Document No.1200A-001
21 June 1990

2

Information Object Modeling Methodology for the Software Technology for Adaptable, Reliable Systems (STARS) Program

Contract No. F19628-88-D-0032

Task IQM15 - Software First Systems Analysis

CDRL Sequence No. 1200A

21 June 1990

Prepared for:

Electronic Systems Division
Air Force Systems Command, USAF
Hanscom AFB, MA 01731-5000

Prepared by:

IBM Federal Sector Division
800 North Frederick Avenue
Gaithersburg, MD 20879

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

[illegible]

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE June 21, 1990	3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE Information Object Modeling Methodology		5. FUNDING NUMBERS C: F19628-88-D-0032
6. AUTHOR(S) W. Ett		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) IBM Federal Sector Division 800 N. Frederick Avenue Gaithersburg, MD 20879		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Electronic Systems Division Air Force Systems Command, USAF Hanscom AFB, MA 01731-5000		10. SPONSORING/MONITORING AGENCY REPORT NUMBER CDRL Sequence No. 1200A
11. SUPPLEMENTARY NOTES		
12a. DISTRIBUTION/AVAILABILITY STATEMENT		12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Information Object Modeling is a technique for developing specification models for systems. The techniques for building Information Object Models were adapted from techniques of real-time structured analysis and the Foxboro company's experience in specifying and developing real-time process control systems.

An Information Object Model (IOM) is organized to provide levels of information for different audiences, so that one document can meet the needs of different people. A mission statement describes the scope of the system. An overview of the system describes the major functional objects. Finally, each functional object is discussed in detail.

The modeling techniques for an IOM use the graphical techniques of real-time structured analysis, including transformation diagrams (data flow plus control flow), state transition diagrams, and entity relationship diagrams. Transformation diagrams, however, are applied in a different manner, representing the communication of objects organized hierarchically rather than a functional decomposition of processes.

This report introduces the IOM methodology, explains what an Information Object Model is, and provides guidance on developing and reviewing diagrams as part of such models. The report also discusses the brief, yet intense history of a government-run experiment using the Information Object Modeling methodology.

14. SUBJECT TERMS STARS, information object model, specification models, real-time structured analysis	15. NUMBER OF PAGES 147
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

Information Object Modeling Methodology
Document Version: 1.3

21 June 1990

William H. Ett

International Business Machines Corporation
Federal Sector Division
Stars Development Department
800 North Frederick Pike
Gaithersburg, Maryland 20879

CDRL 1200A - IOM Methodology Report

Preface

This report represents an amendment to IBM STARS CDRL 1200, the *DoD AAS ATC Structured Specification* (also referred to as the *DoD AAS ATC IOM*). Many of the ideas in this report evolved from discussions I had with my fellow STARS QM15 Phase I team members. The phase I team included the following personnel:

- Mr. John Anderson of the Boeing Company,
- Mr. Dave Campbell of Unisys Corporation,
- Mr. John Downey of Unisys Corporation,
- Mr. William Ett of the IBM Corporation,
- Dr. Jerry White of the Foxboro Company,
- Mr. Dave Wilson of the Boeing Company, and
- Ms. Tammy Wooley of the Boeing Company.

I also want to express my appreciation to my IBM QM15 team who worked hard to understand what a Foxboro IOM was and how to build one. It was the IBM team that figured out how to build an IOM using the IOM functional object allocation approach described in this document. The members of the IBM STARS QM15 Phase II team included the following personnel:

- Mr. William Ett,
- Ms. Barbara Hoeper,
- Ms. Kimberly Lesho,
- Ms. Joanne Piotrowski, and
- Mr. Robert Simonoff.

Abstract

Information Object Modeling is a methodology for developing system specification models. The techniques for building Information Object Models were adapted from techniques of real-time structured analysis and the Foxboro Company's experience in specifying and developing real-time process control systems. This report describes the methodology learned during the government-run experiment, namely STARS Task QM15. The purpose of this experiment was to see whether it was possible to teach the three STARS prime contractors a methodology developed by Dr. Gerald R. White of the Foxboro company for specifying complex systems in a short period of time. After having learned the methodology, the three STARS prime contractors would attempt to apply it to three complex DoD systems to validate the applicability of Foxboro's methodology for specifying complex DoD systems-in-the-large systems. The approach for the task was to form a team from the three STARS prime contractors to learn and apply the Information Object Modeling methodology. The team was to prepare a specification model for a selected DoD system using this methodology. After the three prime contractors learned how to apply the Information Object Modeling methodology, they were to train teams within their own companies to develop specification models for three different DoD systems-in-the-large systems.

The Information Object Modeling methodology is based on a layered model that identifies layers of processing capabilities and roles. The layered model is used as a template for examining and allocating functional objects to their appropriate layers. This allocation of functional objects to layers leads to a hierarchical organization of functional objects and represents a control hierarchy for the proposed system. Organizing functional objects in a hierarchy may be useful in the design of object-oriented Ada-based software systems.

Keywords

- Computer Integrated Manufacturing
- Layered Model
- Object-Oriented Systems Analysis
- Real-Time Structured Analysis
- Specification Modeling
- Structured Analysis
- System Specification

Contents

Figures

Introduction

The purpose of this report is to introduce the Information Object Modeling methodology, to explain what an Information Object Model is, and to provide some guidance in developing and reviewing diagrams for the Information Object Model. This report will also discuss the brief, yet intense history of STARS Task IQM15 from an IBM perspective. STARS Task QM15 was a government-run experiment to answer several questions, namely:

- How do we initiate the software-first life cycle?
- Can an adequate specification for a contemplated large DoD system be prepared in a short period of time?

These questions will be addressed, on the basis of IBM's QM15 experience in the section entitled *Issues Raised from Our Experience with Information Object Modeling* .

Organization

This report is organized into four sections: 1) QM15 History, 2) Information Object Modeling, 3) Building and Packaging the Information Object Model, and 4) Issues Raised from Our Experience with Information Object Modeling.

The first section provides a brief history of the QM15 task. The second section introduces the Information Object Modeling methodology, defines terminology important to the discussion of Information Object Models (IOM), introduces the *White Layered Model*, and provides a simple example of the form of an IOM and the application of the *White Layered Model* with respect to the weapons system of the B-1B bomber. The third section discusses the methodology for building an IOM, introduces the Information Object Model pa :kaging concept and provides guidance for preparing and reviewing Information Object Model transformation diagrams. Finally, the fourth section addresses several issues associated with the Information Object Modeling methodology:

- Is the Information Modeling methodology, based on the *IOM functional object allocation approach* and the *White Layered Model*, a good general purpose modeling technique for developing system specification models?
- What is the role of the IOM in the Software-First Life Cycle methodology?
- What is role of the IOM in the DoD systems procurement process?

The fourth section also presents the conclusions reached from our experience in the use of the Information Object Modeling methodology.

Appendix B of this report presents IOM diagram notation conventions.

QM15 History

Dr. Gerald R. White of the Foxboro Company was approached by Col. Joseph Greene of the STARS program office to explore the possibility of Dr. White's teaching his techniques for analyzing and specifying systems to the three STARS prime contractors. Foxboro had the following interesting capabilities:

- They have developed much expertise in the domain of real-time process control
- They successfully practice software reuse in the development of their systems
- They develop specifications for complex systems in a 90 to 120 day period.

Col. Greene was motivated by Foxboro's track record in the successful application of real-time structured analysis techniques (Foxboro is cited in the acknowledgements of the textbook trilogy *Structured Development of Real-Time Systems* (War-01) by Paul Ward and Steven Mellor). Dr. White's team at Foxboro are experts at the development and integration of complex process management and control systems and they successfully practice software reuse, with their Industrial/Automation series of process control computers. Col. Greene was also motivated by the fact that to win business in the field of industrial process control, the typical procurement cycle takes place over a small time period, in which vendors must understand requirements and develop solutions to complex distributed real-time process control problems. Dr. White indicated to Col. Greene that given a team trained in Foxboro's techniques, Foxboro could develop a complex systems specification in 90 days.

After much discussion, Dr. Gerald R. White agreed to apply his techniques to a DoD program. STARS Task QM15 was established to be executed in four phases:

1. QM15 Task Planning (Dr. White and Boeing Co.)
2. Apply the Information Object Modeling methodology to specify aspects of the B1-B Strategic Bomber; This phase produced:
   - A Foxboro-style Structured Specification (also referred to as an Information Object Model)
   - Personnel trained in Dr. White's Techniques
3. Apply the Information Object Modeling methodology to three DoD systems-in-the-large applications:
   - IBM - Military Air Traffic Control IOM
   - Unisys - Naval Command and Control System/Afloat IOM
   - Boeing - B-1B Implementation Model

     Boeing was given the charter during this phase to learn about Foxboro's Implementation Modeling techniques and to apply them, on the Weapons functional object, as described in the B-1B IOM.
4. Document the methodology and develop a specification modeling environment architecture. This phase was to be performed by the QM15 phase I team. However, this phase was approached differently by the three STARS prime contractors.

Information Object Modeling

Information Object Modeling is the term given to the methodology for developing a specification model for a proposed system. This specification model is intended to be a "what" specification, which employs "how" tools[1] to describe the processing required for a proposed system. In this sense, the Information Object Model is a machine-independent logical design for a target system, which is bound by technology constraints only where it is appropriate to do so. The Information Object Model addresses the essential processing a system must perform and does *not* address processing constraints such as performance requirement constraints, etc.

The notion of an Information Object Model as an essential model was derived from the work of Paul Ward and Steve Mellor as discussed in their textbook trilogy entitled *Structured Development for Real-Time Systems* (War-01). The purpose of the essential model is to separate the "essence" of a system from its implementation details. The essential model for a proposed system will characterize the specific environment in which it must function. The essential model will be described by its essential activities and its essential memory, where essential activities describe what the system must do and essential memory describes what data the system must store. Finally, the essential model is based on the assumption of technology independence. This means we shall define the essential activities the proposed system must perform, unconstrained by how these activities are to be implemented.

The Information Object Modeling methodology employs several techniques for building the specification model. To support the rapid accumulation of knowledge, knowledge engineering-style interviewing techniques are employed. To support the organization of functional objects and the modeling of control and data message communication between them, a layered model is employed to allocate functional objects. Using this layered model, a functional object hierarchy is prepared, based on the roles a functional object plays, along with its processing capabilities. To satisfy the information needs of different types of readers, the specification model is prepared by using an information layering approach. Using this approach for packaging the specification model, overview material is prepared to facilitate an understanding of the purpose and scope of the proposed system. Detailed information is presented to understand the detailed processing requirements for aspects of the system.

The Information Object Model

An *Information Object Model* is the specification model produced using the *Information Object Modeling* methodology. A specification model is the term applied to a specification that provides a logical design of a system to describe its essential requirements. This logical design is not intended to dictate the architecture for a proposed system; this is accomplished in the implementation model, which is the technology-constrained model that addresses implementation constraints. such as performance constraints, human factors engineering, etc. The specification model serves as a processable[2] model (Blu-01) so that it is clearly understood "what" the proposed system is to accomplish. A specification model captures the essential processing a system must accomplish, devoid of implementation technology and performance constraints.

[1] "How" tools refer to graphic-based design tools for describing system aspects of a proposed system. These graphics when packaged together with appropriate descriptions, represent a machine-independent logical design. These graphic-based tools include Ward-Mellor style transformation graphs that illustrate data and control flow, entity-relationship diagrams, and state transition diagrams.

[2] A processable model is one that can be simulated by hand, such that all processing the model is to accomplish can be tested and verified by using selected test cases. This concept was introduced in the paper by Blumofe and Hecht entitled *Executing Real-Time Structured Specifications* (Blu-01).

An Information Object Model is organized to provide different levels of information for its target audiences, such that one document may meet the information needs of different types of people. A mission statement is provided that describe the scope of the system to be addressed by the IOM. An overview of the system provides the reader with a high-level description of the major functional objects of a proposed system. Finally, a detailed discussion of each of the functional objects is provided. The functional objects of the IOM are organized in a layered hierarchy, according to their level of capability and role in the proposed system. Executive and technical management could find brief understandable descriptions of a proposed system to meet their information needs in the mission statement and overview. Technical personnel interested in the details of the proposed system could review the individual functional object descriptions.

Functional Objects versus Objects

For the purposes of this report, we shall define an object to be an encapsulation of characteristics and operations, where:

- All objects with the same characteristics are defined by and belong to an object class
- All objects that belong to an object class are subject to and conform to the same rules, e.g., have the same operations and exhibit the same behavior, given the same stimuli, etc.

All objects can be described by their characteristics, and based on these characteristics, they belong to a particular object class. However, not all objects perform operations. We define objects that do not exhibit any behavior[3] and perform any operations, but are manipulated by other objects as *static objects*. Objects that receive and/or respond to stimuli (messages) and exhibit behavior are referred to as functional objects. Functional objects have a set of operations they perform and are capable of transitioning between several states, on the basis of operations performed by the functional object. The object also has internal data that it employs to perform the data computations and/or data transformations required by an operation. The major difference between static objects and functional objects is that functional objects perform their own operations and effect their own state changes on the basis of external stimuli. In building Information Object Models, we identify, model and allocate functional objects.

To summarize, functional objects have the following characteristics:

- They have one or more potential states
- They have internal data
- They have operations that can receive and/or pass messages to other objects
- They exhibit behavior.

The Two Information Object Model Approaches

In the process of learning the techniques of how to build an Information Object Model, we discovered that there were two approaches that an analysis team could take. These two approaches are:

- The Structured Analysis process decomposition approach
- The IOM functional object allocation approach.

The choice of approach depends on the type of system that is to be modeled. Systems that have the characteristics of data-driven information processing systems are best described by using the Information Engineering methodology, which includes information modeling and structured analysis as two of its steps.

[3] Behavior of an object refers to its ability to change its state based on the triggering of state transition conditions.

Systems that exhibit real-time system characteristics, but are basically information systems can be described by using a combination of information modeling and real-time structured analysis techniques.

Systems that have the characteristics of distributed hierarchical control systems can be described by using the Information Object Modeling methodology.

The following sections describe the two approaches for modeling and preparing an IOM. The major difference between the two approaches is the use of process decomposition versus object allocation to describe the essential processing required of a system.

The Structured Analysis Process Decomposition Approach

The IOM structured analysis process decomposition approach is the application of real-time structured analysis and the use of the IOM packaging concept introduced in this report in the section entitled *Packaging the Information Object Model.* Real-time structured analysis uses of the concept of process decomposition. Using the process decomposition approach to modeling, an analyst will decompose a process into a number of sub-processes. This decomposition of sub-processes will continue from level to level until the process being modeled is decomposed into a set of functional primitives, which collectively describe the work that the process being modeled performs.

The top level in this leveled set of diagrams is called the "context diagram" or level 0. The middle level (diagrams 1 through n) portray the breakdown of some or all processes into a network of processes that can be broken down further. The bottom level consists of a set of functional primitives.

Applying process decomposition requires preparing a hierarchy of diagrams. Each process on a diagram represents a set of encapsulated functions, which are decomposed and represented on a lower-level diagram as illustrated in Figure 1 on page 7.

An example of an Information Object Model prepared by using the *Real-Time Structured Analysis* methodology can be found in:

- *A Reference Model for Computer Integrated Manufacturing* in chapter 4 entitled "The Data-Flow Graph, A Functional Network View of the CIM Reference Model" (Wil-01). Dr. White referred to this model as a "structured analysis IOM."

The IOM Functional Object Allocation Approach

The IOM functional object allocation approach is fundamental to the *Information Object Modeling* methodology described in this report. The Information Object Modeling methodology uses the *White Layered Model* as a template to examine the roles and capabilities assigned to functional objects. After their capabilities and roles are understood, functional objects are allocated to a particular layer of processing capability. The functional objects identified are further organized and partitioned as necessary. Then information and control flow between the functional objects are established. This results in a hierarchy of diagrams, where each functional object on a diagram communicates with subordinate functional objects, peer functional objects, and/or its parent functional object, forming a hierarchy of communicating functional objects that each perform a specific task or set of tasks. This hierarchy of functional objects represents a control hierarchy where functional objects report to parent objects to provide them with information necessary to accomplish their work. Parent objects provide the necessary data and control messages to their subordinate objects and to their peers to direct their work activities. An example IOM diagram hierarchy is illustrated in Figure 1 on page 7.

An example of a document prepared by using the *Information Object Modeling* methodology described in this report can be found in:

- *DoD Advanced Automation System Structured Systems Specification* (also referred to as the *DoD AAS IOM*). IBM STARS CDRL 1200.

The remainder of this report will only address the *Information Object Modeling* methodology based on the IOM functional object allocation approach. A thorough discussion of *Real-Time Structured Analysis* can be found in *Structured Development for Real-Time Systems* (War-01).

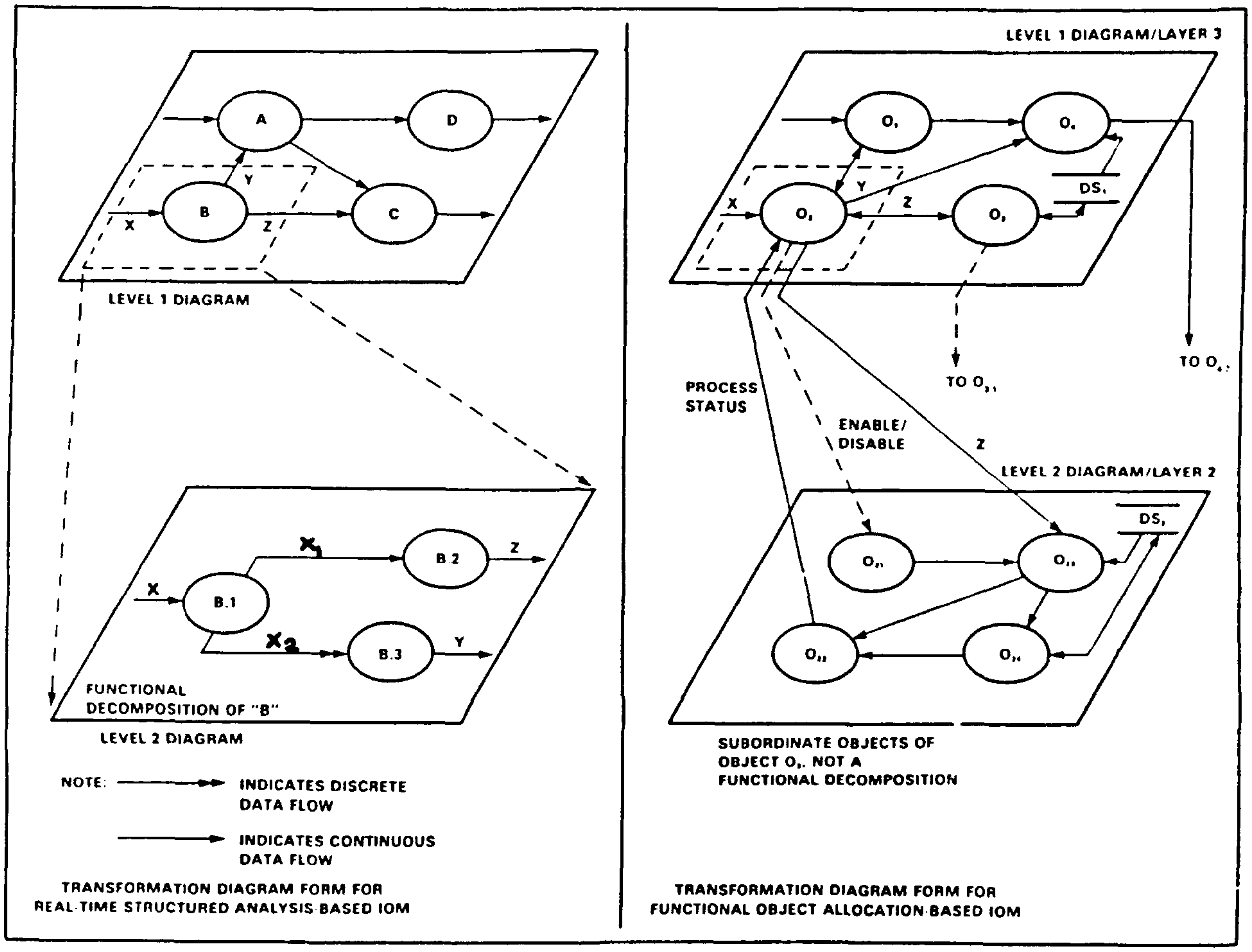

LEVEL 1 DIAGRAM
A
D
B
C
X
Y
Z
LEVEL 2 DIAGRAM
B.1
B.2
B.3
X_1
X_2
FUNCTIONAL DECOMPOSITION OF "B"
NOTE: INDICATES DISCRETE DATA FLOW
INDICATES CONTINUOUS DATA FLOW
TRANSFORMATION DIAGRAM FORM FOR REAL-TIME STRUCTURED ANALYSIS-BASED IOM
LEVEL 1 DIAGRAM/LAYER 3
DS,
TO O,,
TO O,:
PROCESS STATUS
ENABLE/ DISABLE
LEVEL 2 DIAGRAM/LAYER 2
SUBORDINATE OBJECTS OF OBJECT O,. NOT A FUNCTIONAL DECOMPOSITION
TRANSFORMATION DIAGRAM FORM FOR FUNCTIONAL OBJECT ALLOCATION-BASED IOM

Figure 1. The Two Information Object Modeling Approaches. This figure illustrates the basic diagram organization for both *Real-Time Structured Analysis-based IOM* and the *Functional Object Allocation-based IOM*

The White Layered Model

The *White Layered Model* was based on Foxboro's knowledge of process control systems and Computer Integrated Manufacturing (CIM) (Wil-01, Whi-01). The *White Layered Model* originally presented to the QM15 phase I team is similar to the model presented in chapter 3 of the textbook *A Reference Model for Computer Integrated Manufacturing* (Wil-01) entitled "The Generic Duties of a CIM System and Their Expression Via the Hierarchical Form of the Reference Model (Scheduling and Control Hierarchy View) of the System."

The *White Layered Model* provides a general model for allocating objects, according to their roles in the system. Further, the layered model is predicated on the idea of a hierarchical control model. All of the objects allocated to specific layers of capability form a tree of functional objects with communication paths between each of the layers. The layered model as proposed by White seems to be useful for guiding the decomposition of systems that have distributed hierarchical control as their basis. The *White Layered Model* is shown in Figure 2. This model will be discussed within the context of the domain from which it evolved, Computer Integrated Manufacturing. In reviewing the description of the layers, it is important to note the differing roles and capabilities that could be satisfied by devices associated with a particular layer, rather than the details of the layers within the *Computer Integrated Manufacturing* model presented.

```
LAYER 5     - STRATEGIC PLANNING (Corporate management)

LAYER 4     - MANAGEMENT CONTROL (Planning production)

-----------------------------------------------------------------
LINE BETWEEN BATCH AND ONLINE TRANSACTION PROCESSING AND REAL TIME
-----------------------------------------------------------------

LAYER 3     - REAL-TIME DECISION SUPPORT LAYER (Allocating and
              supervising materials and resources)

------------------------------------------------------------------
DIFFERENCES BETWEEN 2, 1.5 AND 1 ARE LEVELS OF DEVICE INTELLIGENCE
------------------------------------------------------------------

LAYER 2     - SUPERVISORY CONTROL LAYER (Coordinating multiple
              manufacturing processes and operations)

LAYER 1.5 - ADAPTIVE CONTROL LAYER  (Commanding device sequences
              and motion of analog devices;  interfaces with special
              purpose sensor devices)

LAYER 1     - LOGIC CONTROL LAYER (Commanding device sequences and
              motion of digital devices;  interfaces with sensors)

LAYER 0     - SENSOR / ACTUATOR LAYER (Activates sequences and motions
              of devices;  senses desired aspect of a manufacturing
              process)
```

Figure 2. The White Layered Model. This figure shows the layers of the White Layered Model and identifies boundaries between capabilities performed in non-real-time systems management and real-time systems management.

The Purpose of the Layered Model

The layered model provides a template for examining functional objects and the roles that they perform. Further it provides a template for the allocation of functional objects, that is based on their processing capabilities. The layered model assists systems engineers/analysts:

- In segregating functional objects based on their degree of complexity and the roles they perform
- By giving clues how to investigate identified functional objects and communication paths between peer, parent or children objects.

Layers 3 through 0 can be represented as a hierarchy of functional objects where the layers have decreasing levels of device intelligence. Functional objects may be spread across layers 3, 4 and 5 as appropriate on the basis of the roles the functional objects perform.

Layers 3 through 0 are used to model systems that exhibit the properties of distributed hierarchic control. Layers 5 through 4 are examined to understand the interfaces between objects identified in these layers and the hierarchic control systems being developed. For example, in modeling the B-1B Strategic Bomber, the LOG functional object of the B-1B interfaced with POST-MISSION ANALYSIS and MISSION OBJECTIVE. These functional objects belong to layer 4 and are part of the MISSION PLANNING functional object. The roles of these functional objects were synthesized to understand their interfaces to the B-1B.

Non-Real-Time Management Layers

The top two layers of the *White Layered Model* are:

- Layer 5 or the Strategic Planning Layer
- Layer 4 or the Management Control Layer.

These are the non-real-time systems management layers.

The Strategic Planning Layer (Layer 5)

The strategic planning layer of the layered model is where corporate strategic planning functions are performed. The information processing requirements to support corporate layer functions are generally satisfied by summary data extracted from Management Information Systems databases. Reporting is usually on a quarterly and monthly basis.

Typical hardware required to support corporate layer needs are satisfied by large mainframe computers, such as the IBM 3090.

The Management Control Layer (Layer 4)

The management control layer of the layered model comprises the enterprise functions of the organization. Planning and scheduling of production are performed and supported by the management information systems function. The information processing requirements to support the management control layer functions are satisfied by summary data, extracted from the real-time management systems databases. Management control layer functions usually can be characterized by transaction-type batch processing, where the MIS reporting systems produce reports on a daily, weekly or monthly basis.

Typical support provided by plant management information systems are:

- Product design and production engineering
- Production management (weekly, monthly)
- Resource procurement (weekly, monthly)

- Resources management (weekly, monthly)
- Maintenance management (weekly, monthly).

Typical hardware required to support management control needs are satisfied by small mainframe computers such as an IBM 4381, IBM 9370, IBM AS/400, IBM System 36/38, DEC VAX 11/78, etc.

The Real-Time Systems Management Layers

The Real-Time Systems Management Layers represent the devices typically employed to design and implement real-time plant process control systems. These layers refer to layers 3 through 0 of the *White Layered Model.* Allocating functional objects to these layers is our primary concern in building Information Object Models for systems with real-time processing requirements. Figure 3 on page 12 illustrates a functional object tree of hierarchically organized functional objects; this figure demonstrates the allocation of objects to layers in the *White Layered Model.*

Real-Time Decision Support Layer (Layer 3)

This layer supports the daily production scheduling and operations of plant manufacturing processes and controls the allocation and supervision of plant materials and resources. Devices at this layer have the responsibility of coordinating manufacturing jobs, as well as obtaining and allocating resources to those jobs.

The real-time decision support layer needs to track the use of raw materials in the manufacturing processes. For example, materials consumption needs to be reported on a timely basis. Monitoring the consumption rate of materials may indicate potential production problems, e.g., reporting that there are insufficient raw materials available to meet the day's expected output.

The real-time decision support layer needs to support statistical quality control to make product quality measurements at specific times in the manufacturing process. Further, on the basis of the product quality measurements taken, the manufacturing process may need to be adjusted.

Example functions assigned to the real-time management layer are:

- Production management (hourly, daily)
- Resource procurement (hourly, daily)
- Resources management (hourly, daily)
- Maintenance management (hourly, daily)
- Shipping
- Waste material treatment.

Typical hardware to support real-time decision support requirements are satisfied by small mainframe computers or super mini-computers such as the IBM 9370, IBM RISC System 6000, MicroVAX, HP-1000, etc.

Supervisory Control Layer (Layer 2)

The supervisory control layer is responsible for coordinating multiple distributed machines and their operations. Distributed control systems sequence and supervise manufacturing processing jobs at a selected factory floor or plant location.

Layer 2 devices are typically off-the-shelf Distributed Control Systems that provide:

- Supervisory control
- Control for special manufacturing process devices
- An interface to Programming Logic Controllers (PLC) and/or Device Multiplexors.

Adaptive Control (Layer 1.5)

The adaptive control layer is responsible for monitoring and reporting equipment sensor readings and for commanding machine sequences and the motion of analog devices. This layer is less capable than a layer 2 device, but can control analog devices. For example, a PLC might be programmed to command an actuator to open or close a valve. However, the process may require a device that can partially open a valve to regulate the flow c. material required and would require a controller capable of analog command processing.

Logic Control Layer (Layer 1)

The logic control layer (formerly referred to as the Programmable Logic Controllers / Multiplexor Layer) is responsible for monitoring and reporting equipment sensors and for commanding machine sequences and the motion of binary devices. Several sensors and actuators may be attached to a layer 1 device for reporting and controlling purposes.

Layer 1 devices typically are digital binary controllers that cause an actuator to put a device into an open state or a closed state. An example of such a device would be a door controller, where the PLC would direct a door actuator to open or close a door depending on the appropriate logic condition.

Sensor/Actuator Layer (Layer 0)

Sensors and actuators are the lowest-level devices represented in the *White Layered Model.* Sensors provide special-purpose readings on aspects of manufacturing processes. For example:

- Fluid flow sensors
- Temperature sensors
- Level sensors
- Pressure sensors.

Sensors provide measurements required to control manufacturing processes.

Actuators and manipulators are controlled by layer 1 and 1.5 devices and are the mechanisms that activate sequences and motion, as required.

```
                             RTM      (1)                       LAYER III
                              |
  +-------------+---------+-------+-------------+-------  ...
  |             |                 |             |
 DCS           DCS               DCS           DCS (2-20)   LAYER II
  |             |                 |             |
  .             |                 .             .
  .             |                 .             .
                |
  +---------+---+-----+----------+ ...
  |         |         |          |
 PLC       PLC       PLC        PLC  (50 - 100)             LAYER I
  |         |         |          |
  .         .         |          .
  .         .         |          .
                      |
       +---------+---+-----+----------+ ...
       |         |         |          |
   ACTUATOR   SENSOR   ACTUATOR    SENSOR  (30 - 50)        LAYER 0
```

Figure 3. The Layered Model Example. This figure illustrates the hierarchical allocation of devices for an industrial process. The connecting lines represent communication paths between the functional objects. Please note: RTM = Real Time Manager, DCS = Distributed Control System and PLC = Programmable Logic Controller.

Applying the Layered Model to the B-1B

To illustrate that the layered model could be extended to other domains, we drew an analogy between the layered model for manufacturing process control and the avionics of the B-1B Strategic Bomber. It is interesting to note that manufacturing plants can be viewed as distributed heterogeneous systems attached to a local area network. Similarly, B-1B avionics systems are heterogeneous systems attached to a MIL-STD-1553 bus. Both typical manufacturing process control systems and the B-1B strategic bomber have a real-time management system that monitors and controls subsystems and issues reports as appropriate.

In examining one of the hierarchic threads of control from the B-1B weapons system, we can derive the following functional object mapping to the layers, as depicted in Figure 4. This hierarchic thread of control is depicted in Figure 5 on page 13.

o	LAYER 3, REAL-TIME MONITORING SYSTEM LAYER	-- WEAPONS CONTROL
o	LAYER 2, DIGITAL CONTROL SYSTEM LAYER	-- LAUNCH CONTROL
o	LAYER 1.5, CONTROLLER LAYER	-- LAUNCH SEQUENCER
o	LAYER 1, PROGRAMMABLE LOGIC CONTROLLER LAYER	-- DOOR CONTROL
o	LAYER 0, SENSOR LAYER	-- WEAPON SYSTEM SENSORS

Figure 4. The B-1B Strategic Bomber Layered Model Example. This figure illustrates a mapping of devices from one of the hierarchic control threads from the B-1B strategic bomber's weapons system.

```
                         Weapons                          (LAYER 3)
                         Control
                            |
... ------------------------+------------------------- ...
                            |
                         Launch                           (LAYER 2)
                         Control
                            |
... ------------------------+------------------------- ...
                            |
                         Launch                          (LAYER 1.5)
                         Sequencer
                            |
... ------------------------+----------------+-------- ...
                            |                |
                         Door               Bomb          (LAYER 1)
                         Control            Release
                            |               Control
                            |                |
... -----+------------------+--...   ...-+--------+-------- ...
         |                  |            |        |
       Door              Door           Bomb     Bomb     (LAYER 0)
       Actuator          Sensor         Release  Rack
                                        Actuator Sensors
```

Figure 5. B-1B Hierarchic Control Thread Mapped to the Layered Model. This figure illustrates the hierarchical allocation of devices for a control thread of the B-1B weapons system. The connecting lines represent communication paths between the functional objects.

IOM Form

In the previous section, we introduced the *White Layered Model* which is of central importance to the Information Object Modeling methodology. This methodology employs the layered model as a template for gauging the capabilities and placement of functional objects in a layered object hierarchy. The purpose of this section is to illustrate and discuss the proper form for a Foxboro-style Information Object Model transformation diagram[4] as shown in Figure 6 on page 15.

Figure 6 on page 15 contains two diagrams which illustrate the following points:

- Diagram 7.1.1 is not a functional decomposition of the functional object (FO) 7.1.1 PRODUCTION RATE CONTROL. FO 7.1.1 PRODUCTION RATE CONTROL does not represent an encapsulation of heaters, temperature sensors, temperature controllers, temperature digitizers, and a pressure sensor. However, diagram 7.1.1 does illustrate the processing performed by a set of subordinate functional objects and the data they must provide FO 7.1.1 PRODUCTION RATE CONTROL for it to accomplish its job.

  FO 7.1.1 PRODUCTION RATE CONTROL receives temperature information from two temperature digitizers (children or subordinate functional objects), and on the basis of data from the PROCESS LIMITS data store, it issues temperature adjustment commands.

[4] The transformation diagram as proposed by *Structured Development of Real-Time Systems* (War-01) shows data flow and control flow between data transformations (processes) and control transforms. These diagrams only illustrate data and control passing between peer processes. An IOM transformation diagram is similar in appearance and in use of notation; however, data and control flow may pass to peer objects with the same parent, parent objects or children objects. This distinguishes the IOM transformation diagram from the Ward-Mellor transformation diagram.

- Notice that there is data and control flow between peer objects sharing the same parent, on diagram 7.1.1, as there is on diagram 7.1
- Notice that there is data flow between diagram levels:
  - FO 7.1.1.2 TEMPERATURE "B" DIGITIZER passes "TEMP B" to FO 7.1.1 PRODUCTION RATE CONTROL. Please note that on diagram 7.1.1, the destination of the data flow "TEMP B" is identified as 7.1.1, which indicates that a message is being sent to FO 7.1.1 PRODUCTION RATE CONTROL. Also note on diagram 7.1, the data flow labeled "TEMP B" has its source identified as 7.1.1.2, which indicates that a message is being sent from FO 7.1.1.2 TEMPERATURE "B" DIGITIZER.

It is important to recognize that these two diagrams illustrate the allocation of functional objects, based on their capability and role. not their decomposition. Figure 6 on page 15 also illustrates hierarchically organized functional objects and the communications established between the Layer 2 FO 7.1.1 PRODUCTION RATE CONTROL and the Layer 1.0 devices that directly communicate with it. It should be apparent to readers somewhat familiar with the diagramming practices of structured analysis that this is not a structured analysis representation. Taking a structured analysis approach, 7.1.1 PRODUCTION CONTROL might be functionally decomposed as follows:

```
7.1.1  PRODUCTION RATE CONTROL
       7.1.1.1  INITIATE REFINING PROCESS
       7.1.1.2  ACQUIRE SET POINTS
       7.1.1.3  MONITOR TEMPERATURE
                7.1.1.3.1  DIGITIZE TEMPERATURE
                7.1.1.3.2  CHECK TEMPERATURE AGAINST SET POINTS
       7.1.1.4  ADJUST TEMPERATURE
       7.1.1.5  TERMINATE REFINING PROCESS.
```

The IOM drawings in Figure 6 on page 15 show data and control flow between functional objects that encapsulate the above functions as their operations.

Please note that Figure 6 on page 15 represents a logical view of production rate control. The diagrams do not demand a physical implementation that looks exactly like this; however Figure 6 on page 15 does require that the temperature be passed to the FO 7.1.1 PRODUCTION RATE CONTROL, which takes action to regulate the temperature during the heating process. This drawing does not dictate "how" the proposed system is to be designed or implemented.

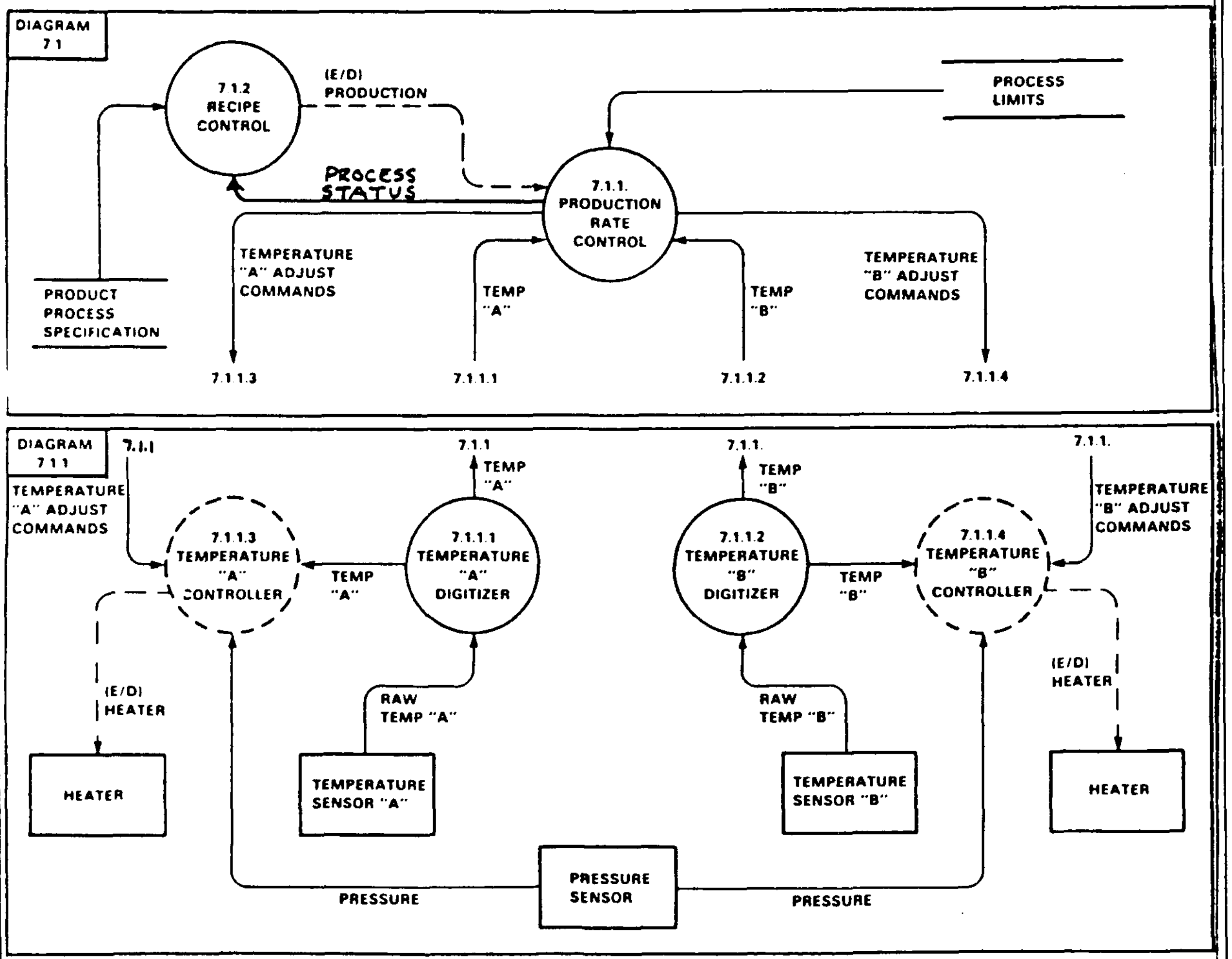


DIAGRAM 7.1
7.1.2 RECIPE CONTROL
(E/D) PRODUCTION
PROCESS LIMITS
PROCESS STATUS
7.1.1. PRODUCTION RATE CONTROL
TEMPERATURE "A" ADJUST COMMANDS
TEMP "A"
TEMP "B"
TEMPERATURE "B" ADJUST COMMANDS
PRODUCT PROCESS SPECIFICATION
7.1.1.3
7.1.1.1
7.1.1.2
7.1.1.4
DIAGRAM 7.1.1
7.1.1
TEMPERATURE "A" ADJUST COMMANDS
7.1.1.3 TEMPERATURE "A" CONTROLLER
TEMP "A"
7.1.1.1 TEMPERATURE "A" DIGITIZER
7.1.1.2 TEMPERATURE "B" DIGITIZER
TEMP "B"
7.1.1.4 TEMPERATURE "B" CONTROLLER
TEMPERATURE "B" ADJUST COMMANDS
(E/D) HEATER
HEATER
RAW TEMP "A"
TEMPERATURE SENSOR "A"
RAW TEMP "B"
TEMPERATURE SENSOR "B"
PRESSURE SENSOR
PRESSURE

Figure 6. Production Rate Control IOM. This figure illustrates the proper form for a Foxboro-style IOM. It illustrates the communication of peer functional objects with the same parent, and communication between parent and children functional objects.

Building and Packaging the Information Object Model

Building an Information Object Model is accomplished in several phases, namely:

1. The Information Object Modeling effort planning phase
2. The Information Object Modeling phase
3. The review and acceptance phase.

This section provides a brief introduction to these phases of building an Information Object Model, the guidelines for packaging[5] the information Model and guidance for preparing and reviewing Information Object Model diagrams.

Planning the Information Object Modeling Effort

This section will discuss the activities necessary to plan for a successful Information Object Modeling effort.

Obtain Management Commitment

During the planning phase, the scope of the analysis to be performed is assessed and management commitment is obtained. Before undertaking an Information Object Modeling effort, the customer must be committed to making his valued personnel resources available to the IOM analysis team. Support is needed from the appropriate levels within the corporation to break down barriers to permit the timely and effective collection of information. Without this cooperation, efforts to ascertain the processing requirements for a proposed system, may lead to the specification of a system that the customer does not want.

It is important to understand the roles people currently play in the operation of a system, and the roles people will play in the future. It is important to understand what new roles people will and will not accept in planning a new system. Without proper access to the appropriate corporate personnel involved in a current operation, this understanding may not be achieved.

Prepare the Mission Statement

Given the appropriate management commitment, a mission statement is drafted to describe the mission of the proposed system and establish the boundaries for the analysis. This statement is used to plan interview schedules for corporate personnel to ensure that interviews are properly planned in advance and that optimum use will be made of people's valuable time. The mission statement serves as a document of understanding between the analysis team and corporate project management that commissioned the IOM. It provides the scope of the analysis task and identifies what the IOM should address.

After the mission statement has been prepared and accepted, the remaining planning activities include:

- Selecting the IOM analysis team
- Establishing project standards and conventions
- Identifying activities and preparing an IOM development schedule
- Collecting available documentation
- Identifying domain and system experts

[5] By packaging, I am referring to the organizing of IOM diagrams, supporting graphics and textual descriptions into an Information Object Model.

- Scheduling initial target interviews
- Conducting an analysis effort kickoff meeting.

The planning period typically takes two weeks to two months, depending on the size and complexity of the proposed system. This time period is not included in the 90 to 120 day time estimate for modeling and preparing the IOM, performed during the basic modeling process.

Select the IOM Analysis Team

Team members should be selected based on their general expertise and competence. Familiarity with aspects of the problem domain is helpful, but experts in the problem domain should be avoided. Experts too familiar with the problem domain on the analysis team may have a tendency to dictate solutions to the team, before the team is ready to deal with them.

Team members should be trained in the methodology and tools before the analysis effort begins.

The analysis team should comprise a team leader, system engineers and analysts, and a project documentation and database administrator. There should be sufficient analysts to model their assigned functional objects. The team leader serves as a product reviewer and team facilitator, and prepares for conducting customer reviews, as well as reporting project status to management. The project documentation and database administrator is responsible for collecting and making available system documentation and analysis team work products. He or she is also responsible for model management and integration, whether models are hand drawn or prepared by using Computer-Aided Software Engineering (CASE) tools. A typical IOM analysis team would have 6 to 8 people, depending on the scope and complexity of the IOM. Smaller IOM analysis teams are preferred where it is practical.

Establish Project Standards and Conventions

Team members should review and discuss the tools and techniques to be employed during the analysis effort and should come to a consensus on standards and conventions. This discussion fosters ownership and compliance. Having standards and conventions handed to an analyst with the words, "You shall follow this," may lead to conflicts that may affect the dynamics of the analysis team.

Where existing standards and conventions exist for prescribing the form and style of documentation and graphics, the group should review them. Problems in the existing standards and conventions should be identified and reviewed. If justified, a waiver to the problem sections of the standards should be sought.

Identify Activities and Prepare IOM Development Schedule

The analysis effort should be charted out. Informal and formal reviews should be scheduled and tentative development schedules set. The development schedule will be revised after assignments for modeling functional objects have been made among the analysis team.

Collect Available Documentation

Due to the hectic pace of an IOM modeling effort and preparation required for interviews, it is helpful to have as much available documentation that describes the existing system as possible. This assignment should be given to the team documentation and database administrator.

Identify Domain and System Experts

In initial discussions with corporate and line management, experts with skills, roles and knowledge important to the analysis effort will be identified.

Schedule Initial Target Interviews

Interviews with domain and system experts will be scheduled to assist in the initial modeling performed during the basic modeling process. During these interviews, additional sources of expertise will be identified and added to the "expert" list, and subsequent interviews will be scheduled during the planning of how to analyze and model each functional object.

Conduct an Analysis Effort Kickoff Meeting

After the initial interviewees have been identified, a meeting should be held to explain the goals and scope of the analysis effort. Appropriate customer management and technical personnel should be asked to attend and speak, if appropriate. Holding this meeting demonstrates management commitment to the analysis effort and improves the cooperation the analysis team will receive from customer personnel.

The Information Object Modeling Phase

Information object modeling is a process of stepwise refinement, where model drawings and supporting text are prepared, critiqued and corrected or refined as appropriate. The Information Object Model is complete when it is sufficient to be used for: 1) incrementally developing system prototypes or 2) building an Implementation Model.

Where the target system to be built is an unprecedented system (no system model exists for the system) or where the domain is unfamiliar to the analysis team, the analysis team may elect to build a generic IOM. Building a generic IOM may or may not be the proper thing to do, depending on the mission statement prepared and customer's requirements. The obvious advantage to building a generic IOM is that it serves as a means for building a model that satisfies a generic problem statement, and thus leads to potential model reuse. This is useful in providing the analysis team with knowledge of the problem domain, which will be beneficial in preparing an application-specific IOM.

Where the target system to be built is an unprecedented system, building a generic IOM could be used as a technique to perform a cursory domain analysis of the problem domain, where objects of that domain are identified, described and refined. However, the generic IOM is not a substitute for a formal domain analysis where object classes are established on the basis of common characteristics and operations that a set of member objects possesses.

In summary, the generic IOM provides the systems engineer/analyst with:

- An overview of a complex system
- A model of the major functional objects of a system
- A model of the relationships between functional objects of a system
- A vehicle to understand the application domain.

The QM15 phase I IOM analysis team employed this approach during the first 45 days of the B-1B IOM analysis effort. The IBM QM15 phase II IOM analysis team employed this approach during the first 45 days of the Military Air Traffic Control effort. From our experience, this technique was beneficial in gaining knowledge of the air traffic control problem domain in a reasonably short period of time. However, an extension period of 45 to 60 days should be given to the analysis team to build the generic IOM, and should be considered separately from the 90 to 120 day IOM analysis effort.

Information Collection

Information helpful in building the essential model of a system can be found from a number of sources. These include:

- Searching through existing documentation. In many cases, however, existing documentation may be non-existent or over-abundant, in which case interviewing personnel knowledgeable in a current system and its functional aspects and operations will be essential. When there is an abundance of documentation, a person knowledgeable in the organization of the documentation can help analysts construct a roadmap of how to approach the documentation and will save much valuable time.
- Interviewing personnel knowledgeable in a current system and its functional aspects and operations.

People with expert knowledge fit into two categories:

- Domain experts - people with academic and professional experience in the target domain which affects the system to be built
- System experts - people with professional experience in maintaining and operating a system which is to be upgraded or replaced.

Both categories of people can provide valuable information. The systems engineer/analyst must plan for and properly conduct the interview to satisfy his or her information gathering goals.

The Basic Modeling Process

The basic modeling process consists of 9 steps:

1. Establish the system context for the Information Object Model
2. Identify the major functional objects for the system
3. Identify information pipes between the major functional objects (MFOs)
4. Identify commonly shared data sources
5. Prepare the Information Object Model system overview
6. Identify and allocate subordinate functional objects (FOs) to each major functional object
7. Model the system aspects of the functional objects at each diagram level
8. Model message communication of functional objects between IOM diagram levels
9. Continue diagram decomposition of steps 6 through 8 until layer 0 devices are modeled.

The IOM will be incrementally reviewed and refined during this process, as appropriate. The modeling of the major functional objects, once identified, can be performed concurrently. The above steps introduce activities that must be performed. However, as one gains information during interviews or from reviewing documents, the layered model provides one with a reference model for dealing with information as it is received. Consequently, some steps may have to be rearranged, based on the order and level of information the analysis team receives.

The above steps represent a logical order for introducing activities that must be performed during the analysis effort and a logical order to follow during the modeling process, if it is possible to do so.

1 - Establish the System Context for the IOM

The analysis team needs to establish the context for the proposed system and to identify the external entities (objects, organizations or other systems) that will share, provide or receive messages (data) with the proposed system. This is represented by a context diagram, which is also referred to as the "Level 0 diagram." This diagram shows the system level functional object and its external interfaces. The rules for preparing this diagram are similar to those for preparing a Structured Analysis Context Diagram. An example IOM context diagram is illustrated in Figure 7 on page 21.

In preparing the context diagram, examine the interfaces identified. Avoid representing hardware devices as interfaces. Generalize the role of a hardware device that is providing data to the system, and represent that generalization as the interface on the context diagram. For example, in Figure 7 on page 21, the interface AIR TRAFFIC provides the AREA CONTROL FACILITY with raw radar reports (traffic surveillance information). Traffic surveillance radars were not identified on the context diagram, as they are to be shown at the layer within the IOM that they provide raw radar reports. The surveillance radars appear in the ATC IOM as interfaces on the 1.1.1 diagram as shown on Figure 11 on page 29.

Where the interfaces identified on the diagram have no convenient generalization, it is permissible to include them on the context diagram. As shown in Figure 7 on page 21 there are three forms of weather data, which are provided by different sources. The interface identified as WEATHER is the analog to AIR TRAFFIC. Weather surveillance radars are shown on lower levels of the diagram. WEATHER AGENCY provides WEATHER REPORTS and is employed directly by the air traffic controllers. The REAL-TIME WEATHER PROCESSOR provides weather data directly to FO 2.2 WEATHER TARGET PROCESSING of the FO 2.0 WEATHER SURVEILLANCE. This delineation of sources of weather information should be shown on the context diagram because the AREA CONTROL FACILITY is interfacing with several systems that provide weather information in different forms to satisfy different purposes.

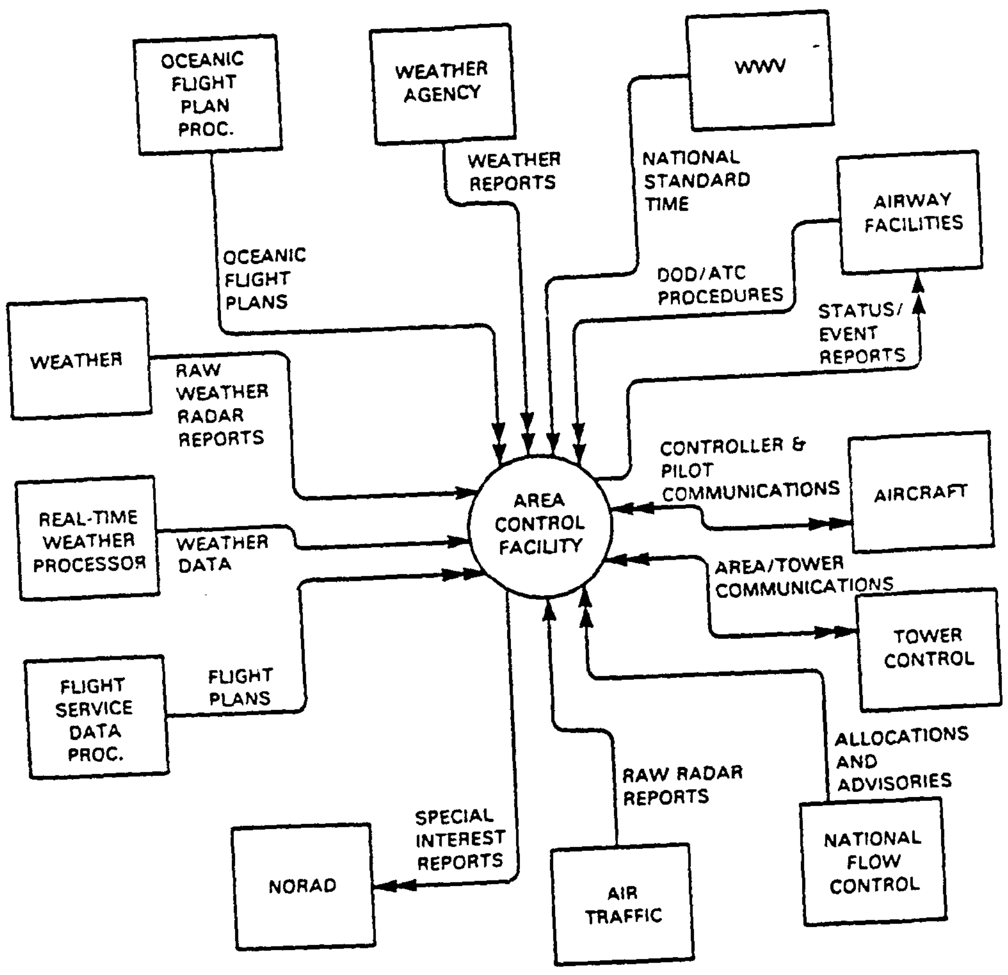


Figure 7. IOM Context Diagram. This figure illustrates an example IOM context diagram from the DoD AAS ATC IOM.

2 - Identify the Major Functional Objects for the System

The analysis team needs to identify the major functional objects for the proposed system, that satisfy the needs for a system stated in the mission statement. These objects will typically be layer 3 objects, that exhibit control over subordinate objects and possibly over peer objects. Layer 5 and layer 4 objects are typically not modeled in a Foxboro IOM because they do not typically conform to a model of hierarchic control. Layer 3 objects typically interface with systems represented in layers 5 and 4, but layer 5 and 4 objects are not covered in a typical Foxboro-IOM.

The major functional objects are determined from comparing the results of several interviews from different sources. Interviewees are asked to identify the major functional areas of a system and asked to describe some of the functions attributed to the major functional areas. From examining the results, a set of major functional objects is formed. Functional objects are established, based on the operations they perform, the data they require and the information they produce.

The major functional objects identified represent the first level of decomposition and show the functional objects encapsulated in the target system as illustrated in Figure 8 on page 23. In structured analysis (Dem-01) and the Information Modeling Methodology, the first-level diagram is referred to as the level 1 diagram[6] . The level 1 diagram shows the functional objects for the proposed system. All functional objects that appear on the level 1 diagram are, by definition, *major functional objects*. Figure 8 on page 23 illustrates that the *DoD AAS Area Control Facility* comprises the following functional objects:

- 1.0 Traffic Surveillance
- 2.0 Weather Surveillance
- 3.0 Prediction and Resolution
- 4.0 Recording Support
- 5.0 Aircraft Track Management
- 6.0 Flight Plan Entry Support
- 7.0 Flight Plan Operation Support
- 8.0 Area Control.

The system level functional object identified in Figure 7 on page 21 as the AREA CONTROL FACILITY is the only object that is typically permitted to be a "hollow bubble."[7]

[6] Diagrams for the IOM are organized in terms of levels. One or more diagram levels can be used to describe an IOM layer.

[7] The term "hollow bubble" is used to describe a convenient encapsulation of functional objects or functions, where the parent bubble is just a hollow shell and does not perform work, e.g. transforms no data, does not control actions of peer and subordinate objects, etc.

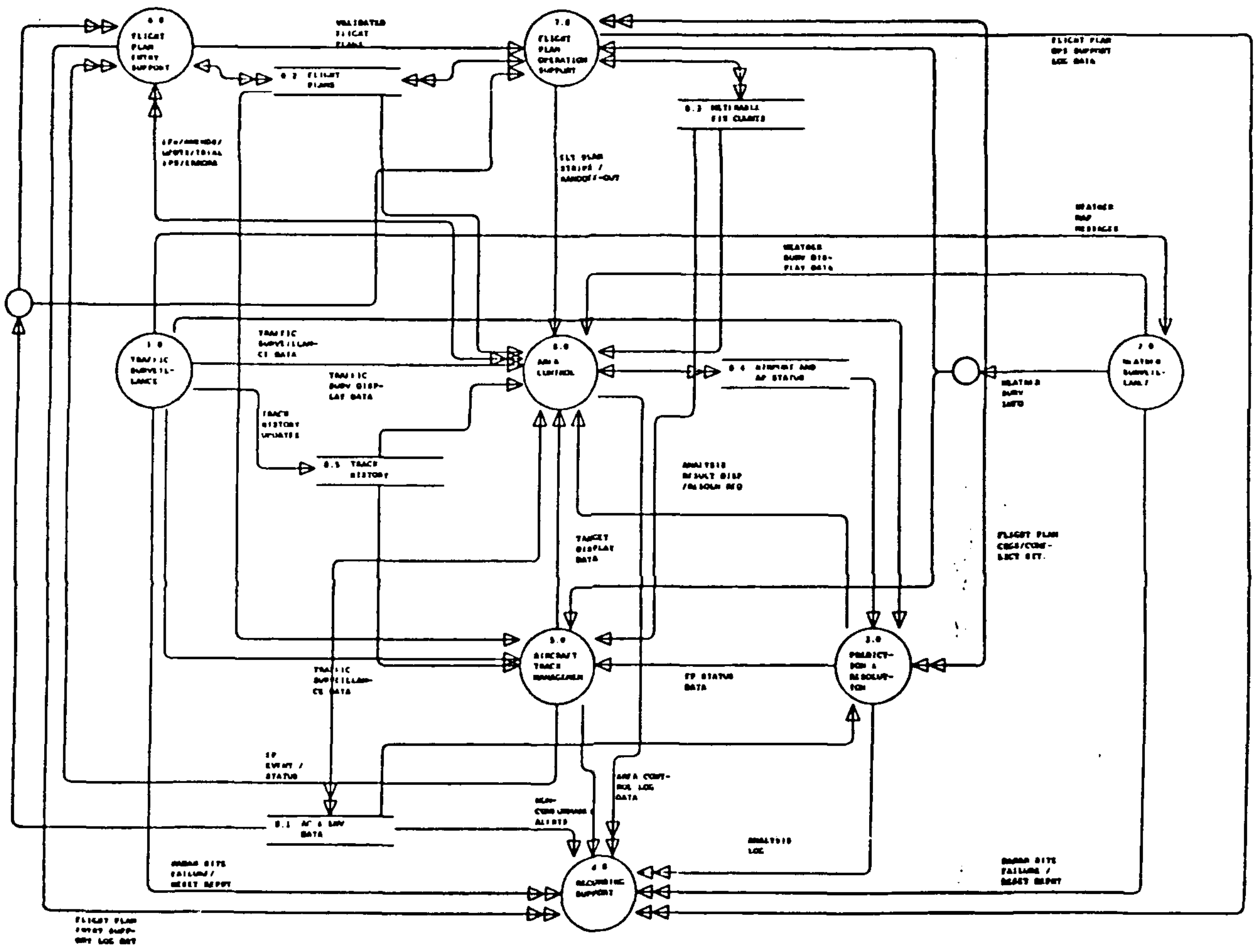
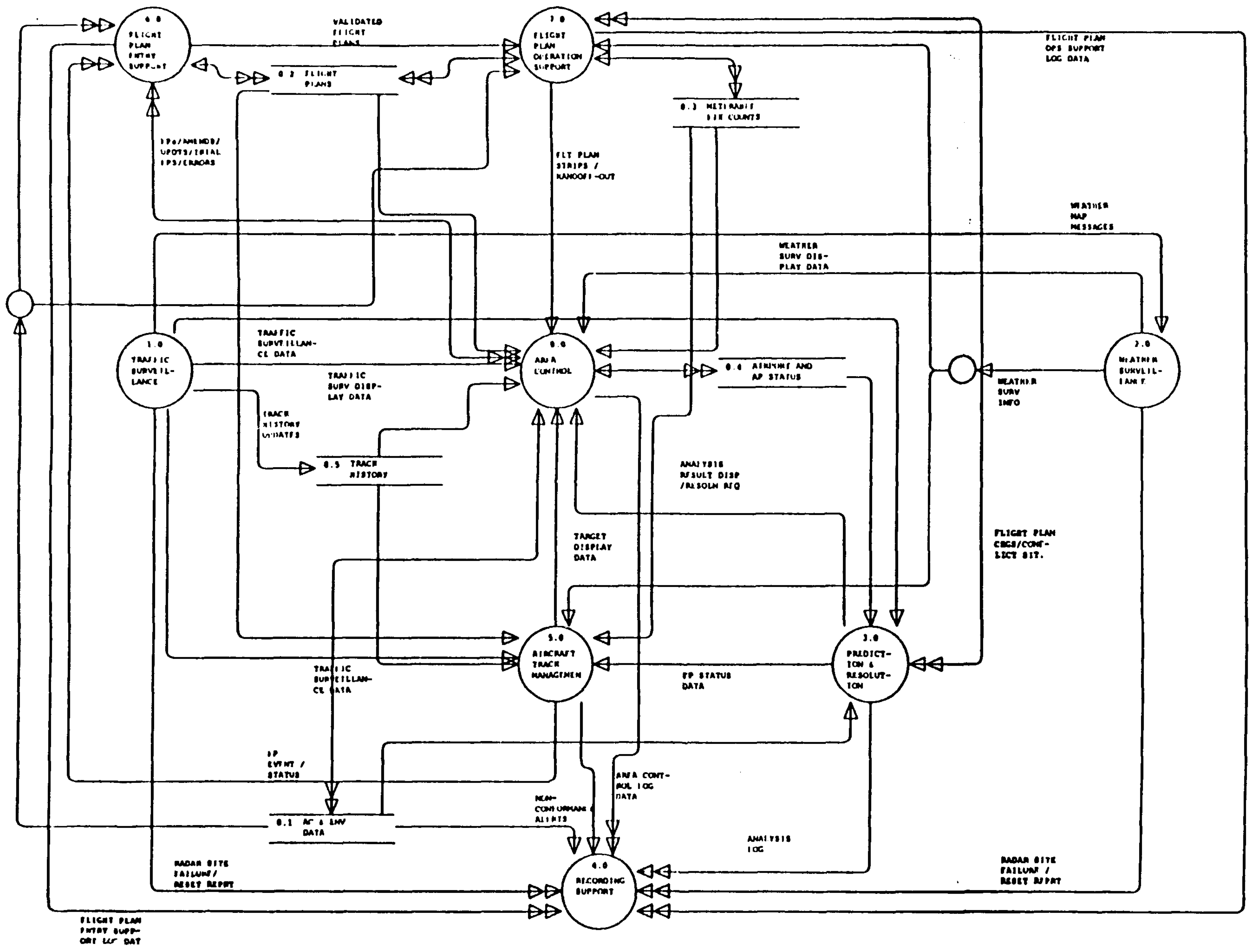


Figure 8. IOM Level 1 Diagram. This figure illustrates an example IOM Level 1 diagram from the DoD AAS ATC IOM. This diagram illustrates the major functional objects of the DoD AAS Area Control Facility, the information pipes that connect them and the global data stores that they employ or maintain.


6.0 FLIGHT PLAN ENTRY SUPPORT
7.0 FLIGHT PLAN OPERATION SUPPORT
VALIDATED FLIGHT PLANS
FLIGHT PLAN OPS SUPPORT LOG DATA
0.2 FLIGHT PLANS
0.3 METERING FIX COUNTS
FPs/AMENDS/ UPDTS/INITIAL FPS/ERRORS
FLT PLAN STRIPS / HANDOFF-OUT
WEATHER MAP MESSAGES
WEATHER SURV DISPLAY DATA
1.0 TRAFFIC SURVEILLANCE
8.0 AREA CONTROL
2.0 WEATHER SURVEILLANCE
TRAFFIC SURVEILLANCE DATA
TRAFFIC SURV DISPLAY DATA
0.4 AIRSPACE AND AP STATUS
WEATHER SURV INFO
TRACK HISTORY UPDATES
0.5 TRACK HISTORY
ANALYSIS RESULT DISP /RESOLN REQ
TARGET DISPLAY DATA
FLIGHT PLAN CHGS/CONFLICT SIT.
5.0 AIRCRAFT TRACK MANAGEMENT
FP STATUS DATA
3.0 PREDICTION & RESOLUTION
TRAFFIC SURVEILLANCE DATA
FP EVENT / STATUS
AREA CONTROL LOG DATA
NON-CONFORMANCE ALERTS
0.1 AC & ENV DATA
ANALYSIS LOG
RADAR SITE FAILURE/ RESET REPORT
RADAR SITE FAILURE / RESET REPORT
4.0 RECORDING SUPPORT
FLIGHT PLAN ENTRY SUPPORT LOG DAT

DoD AAS Area Control Facility. This figure illustrates the major functional objects of the DoD AAS Area Control System. The figure also illustrates the information flows between these objects.

3 - Identify Information Pipes between the Major Functional Objects

The analysis team needs to identify which functional objects share, provide or receive messages (data), and record any relevant attributes that can be associated with the one or more message types. It should be noted that cross communication between functional objects takes place only on the level 1 diagram, as illustrated in Figure 8. The rules for communications between functional objects state that peer objects of the same parent may share peer-to-peer object communications. The subordinate (or children) functional objects of a major functional object may communicate only with its parent, peer subordinate objects that share the same parent, or its subordinate (or children) functional objects. For example, the FO 1.0 TRAFFIC SURVEILLANCE provides TRAFFIC SURVEILLANCE DATA to FO 3.0 PREDICTION AND RESOLUTION and FO 5.0 AIRCRAFT TRACK MANAGEMENT. This is the only level where these functional objects communicate. Subordinate functional objects of FO 1.0 TRAFFIC SURVEILLANCE are not allowed to communicate directly with subordinate functional objects of FO 3.0 PREDICTION AND RESOLUTION and FO 5.0 AIRCRAFT TRACK MANAGEMENT. All communication performed between the major functional objects is only shown on the level 1 diagram.

The level 1 diagram, sometimes referred to as the "spaghetti diagram" because it is difficult to read, shows the large number of information pipes through which information is sent to or received by the major functional objects. To reduce the complexity of presentation on the level 1 diagram, a subset of it is prepared using the "View-From" diagram technique, illustrated in Figure 9 on page 25. The "View-From" diagram is prepared to establish a focus on a single major functional object and to present the reader with only the information relevant to the selected object. One "View-From" diagram is prepared for each major functional object that appears on the level 1 diagram.

To prepare a "View-From" diagram, the major functional objects are organized on a sheet of paper and attention is focused on a selected functional object. Information flows between the other major functional objects are suppressed except for information they send to and receive from the selected functional object. The "View-From" diagram is an excellent tool to focus an interviewee or IOM reader on aspects of a particular major functional object.

After preparing the "View-From" diagram, the IOM "Interfaces" diagram should be prepared. The IOM "Interfaces" diagram combines information from the context diagram and the "View-From" diagram and is used to show the external and internal interfaces to the subject major functional object. An example of an IOM "Interfaces" diagram is illustrated in Figure 10 on page 26. One IOM "Interfaces" diagram is prepared for each major functional object.

The "View-From," "Interfaces" and the "Functional Object Tree" diagrams are used to introduce the detailed description section of each major functional object. This is discussed in the section of this report entitled *Packaging the Information Object Model.* The Functional Object Tree will be discussed later in this report.

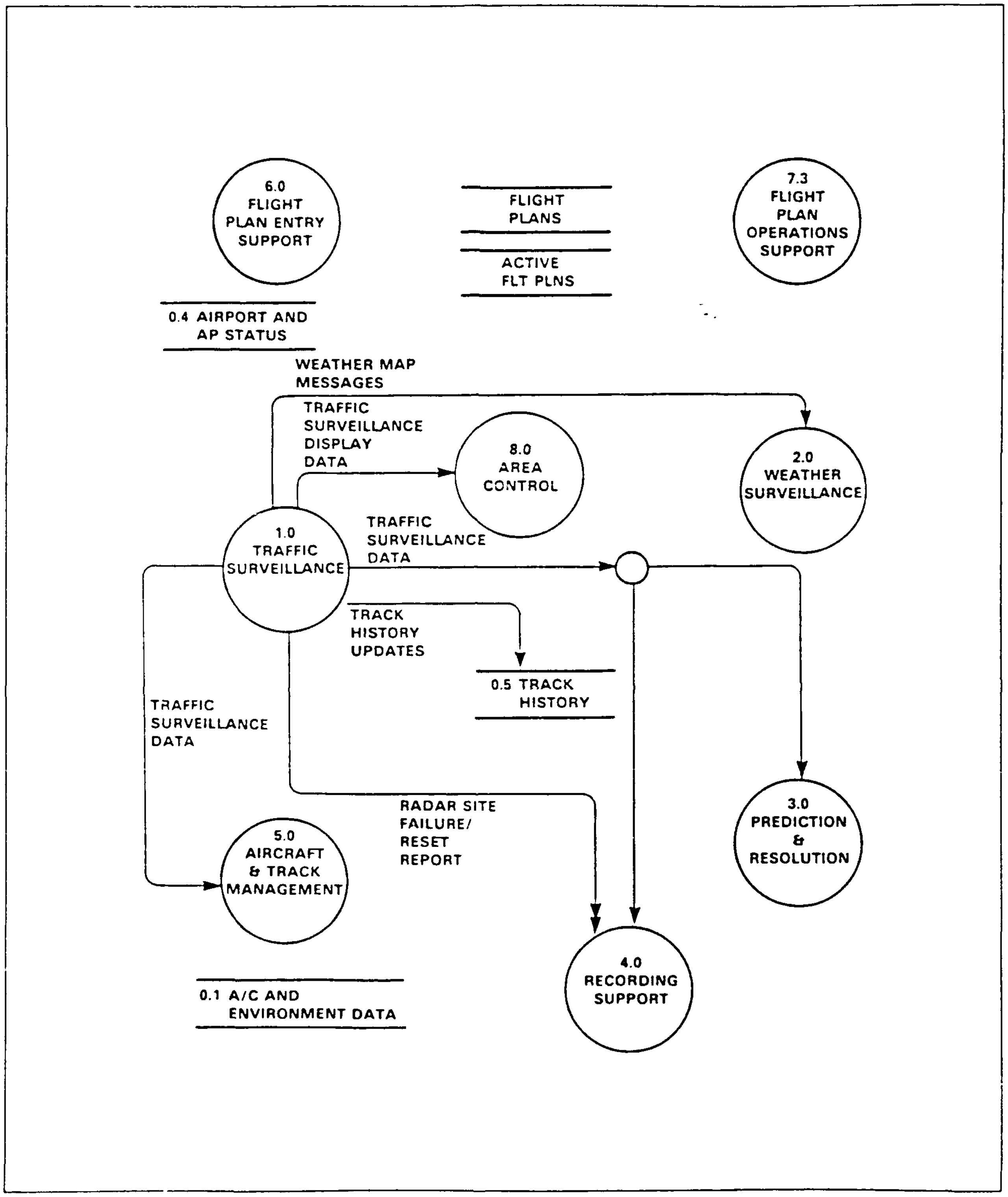

6.0 FLIGHT PLAN ENTRY SUPPORT
FLIGHT PLANS
ACTIVE FLT PLNS
7.3 FLIGHT PLAN OPERATIONS SUPPORT
0.4 AIRPORT AND AP STATUS
WEATHER MAP MESSAGES
TRAFFIC SURVEILLANCE DISPLAY DATA
8.0 AREA CONTROL
2.0 WEATHER SURVEILLANCE
1.0 TRAFFIC SURVEILLANCE
TRAFFIC SURVEILLANCE DATA
TRACK HISTORY UPDATES
0.5 TRACK HISTORY
TRAFFIC SURVEILLANCE DATA
RADAR SITE FAILURE/ RESET REPORT
3.0 PREDICTION & RESOLUTION
5.0 AIRCRAFT & TRACK MANAGEMENT
4.0 RECORDING SUPPORT
0.1 A/C AND ENVIRONMENT DATA

Figure 9. IOM Level 1 "View From" Diagram. This figure illustrates an IOM Level 1 "View-From" diagram from the the DoD AAS ATC IOM. This diagram illustrates the major functional objects of the DoD AAS Area Control Facility, but focuses attention on a selected functional object and the information pipes that it uses to receive and pass messages to other peer functional objects, as well as the global data stores that it employs or maintains.

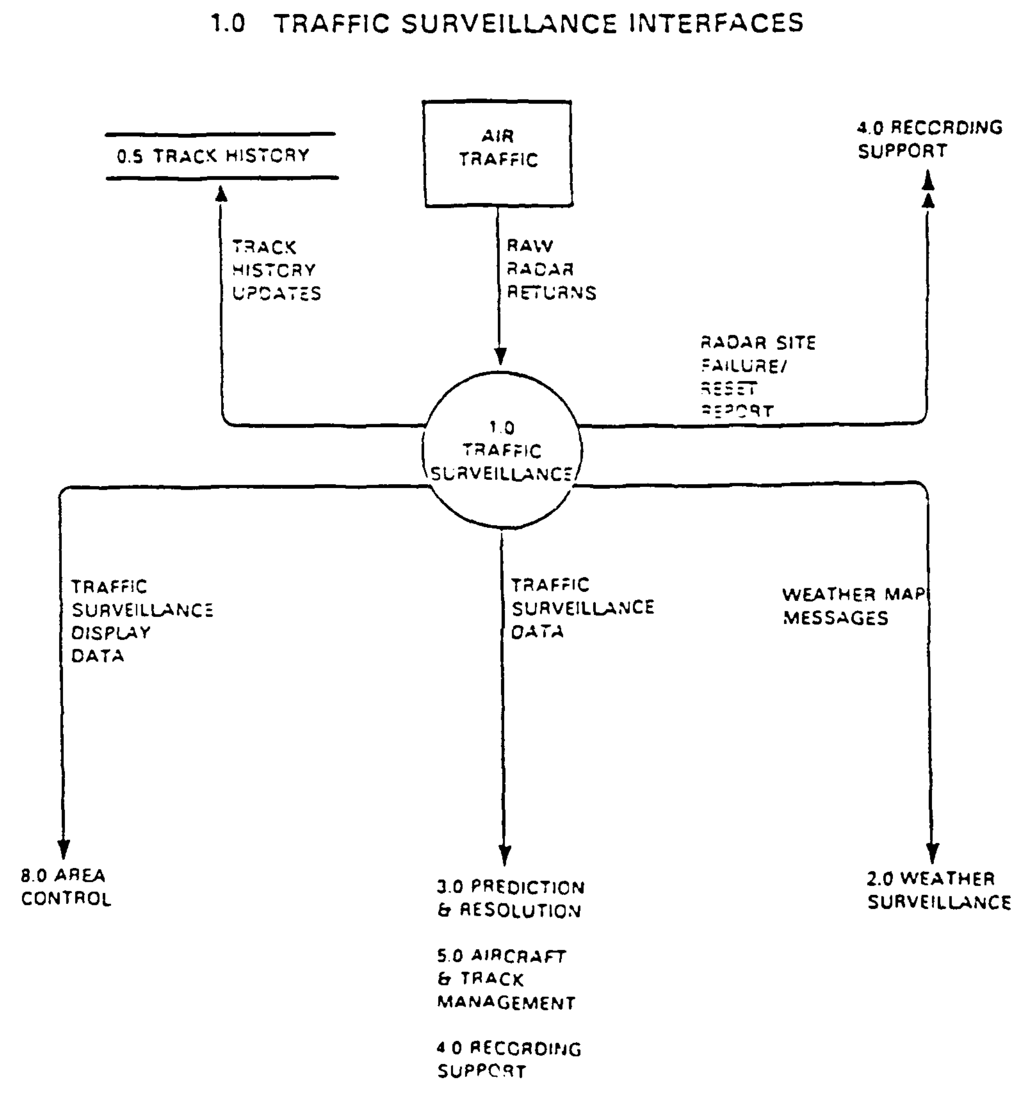


Figure 10. IOM 'Interfaces' Diagram. This figure illustrates an IOM 'Interfaces' diagram. The IOM 'Interfaces' diagram combines information from the context diagram and the 'View-From' diagram and is used to show the external and internal interfaces to the subject major functional object. This diagram is the IOM 'Interfaces' diagram for the major functional object 1.0 TRAFFIC SURVEILLANCE from the DoD AAS ATC IOM.

4 - Identify Commonly Shared Data Sources

During the investigation of the information functional objects share, the analysis team needs to identify global data stores that are derived, updated or shared by more than one functional object. This is illustrated in Figure 8 on page 23. Note that on the level 1 diagram for the DoD AAS Area Control Facility, the data stores FLIGHT PLANS, METERABLE FIX COUNTS, TRACK HISTORY, AIRCRAFT AND ENVIRONMENT DATA, and AIRPORT AND AP STATUS are shown because they are derived, updated or shared by more than one functional object.

Data stores employed by only one functional object are shown in the lower level diagrams describing that object.

5 - Prepare the System Overview

After a context diagram has been prepared for the system, the major functional objects have been identified and information pipes between them have been established, a system overview should be prepared. This overview will be used to validate the major functional objects selected. Further, the overview will be used as a checkpoint with the customer's project management and staff to make sure the analysis team is addressing the system within the boundary of the mission statement. Further, it will be used to ensure that the major requirements for the system are understood by the IOM analysis team and by management.

The management review of the system overview can be a formal or an informal review, depending on the size and scope of the system. The first pass system overview would usually be prepared at the end of the first two to three weeks.

Every member of the analysis team participates in the modeling of the context diagram and the first-level diagram, which illustrates the major functional objects and their information pipes. This provides each team member with a "system" view of the system they are to model and a rudimentary understanding of interfaces between the functional objects. However, after the functional objects have been identified and agreed upon, they are allocated to individual team members for modeling.

After team members are assigned their functional objects, the IOM development schedule is updated to include schedules for their major functional object modeling responsibilities.

6 - Identify and Allocate Subordinate Functional Objects to each MFO

Analysts must identify subordinate (children) objects of each major functional object. These subordinate functional objects are allocated to the appropriate layers of the layered model, based on their system roles and object capabilities. This process of object identification and allocation is performed by using a process of stepwise refinement. An example of the allocation of functional objects is illustrated in Figure 11 on page 29 for the *Traffic Surveillance* functional object of the *DoD AAS ATC IOM*. This figure is referred to as the *Functional Object Tree* (FOT). The Functional Object Tree diagram identifies all the subordinate functional objects of a major functional object and shows parent-child object communications. Please note in Figure 11 on page 29 that FO 1.1.1.1 DIGITIZED RADAR REPORTS provides radar reports to the functional objects in the area labeled diagram 1.1. Where a layer 0 or layer 1 functional object is providing a common service and the functional object is not controlled by any of the upper-level objects, it is permissible to represent the functional object as a common service object, as is shown in Figure 11 on page 29. However, if FO 1.1.1.1 was being controlled by one of the functional objects in the area labeled diagram 1.1, then the hierarchical representation should be enforced.

It is important to note that one layer of the layered model may be represented by several diagram decomposition levels. Although the identified functional objects may properly belong to a *layered model* layer, their role in the control hierarchy being modeled may require multiple diagram levels to model a single IOM layer properly. Ideally, it would be helpful if all functional objects could be modeled within the existing layers of the layered model. Although a set of functional objects may satisfy the membership criteria for a particular layer, several levels may be required to model the functional objects, on the basis of their role within the layered model of hierarchical control. Thus, multiple levels may be required to model a *layered model* layer.

An example of a single processing thread from the functional object tree of the B-1B IOM is illustrated in Figure 18 on page 44. This diagram identifies *Layered Model* layers and levels of decomposition between the layers. Figure 18 on page 44 illustrates that FO 2.5 WEAPONS CONTROL and FO 2.5.1 WEAPONS LAUNCH CONTROL of the B-1B weapons system are layer 2 functional objects. However, FO 2.5.1 WEAPONS LAUNCH CONTROL is subordinate to FO 2.5 WEAPONS CONTROL. As indicated in the figure, a diagram is prepared for each.

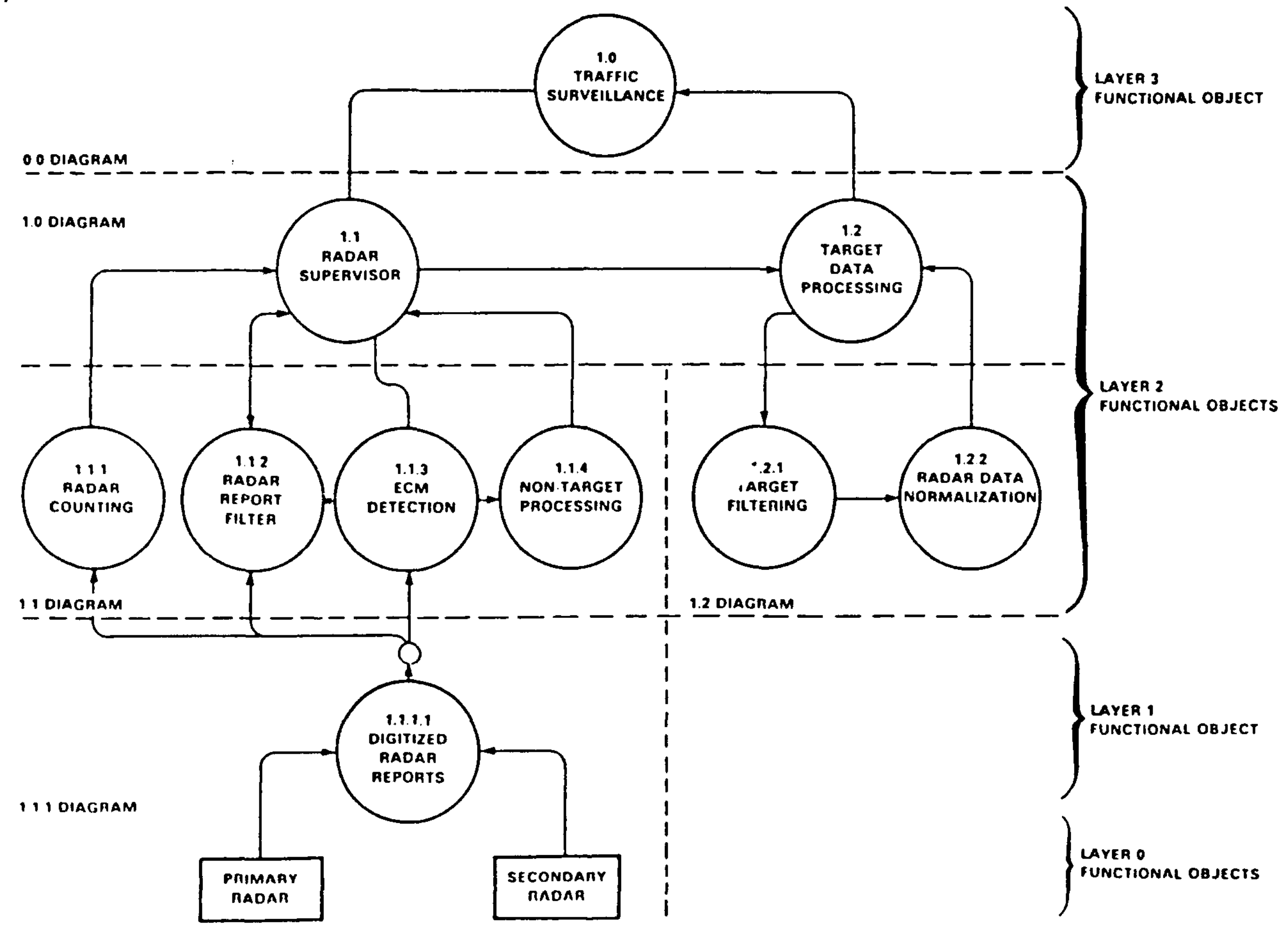

1.0 TRAFFIC SURVEILLANCE
LAYER 3 FUNCTIONAL OBJECT
0 0 DIAGRAM
1.0 DIAGRAM
1.1 RADAR SUPERVISOR
1.2 TARGET DATA PROCESSING
LAYER 2 FUNCTIONAL OBJECTS
1 1 1 RADAR COUNTING
1 1 2 RADAR REPORT FILTER
1.1.3 ECM DETECTION
1.1.4 NON-TARGET PROCESSING
1.2.1 TARGET FILTERING
1.2 2 RADAR DATA NORMALIZATION
1 1 DIAGRAM
1.2 DIAGRAM
1.1.1.1 DIGITIZED RADAR REPORTS
LAYER 1 FUNCTIONAL OBJECT
1 1 1 DIAGRAM
PRIMARY RADAR
SECONDARY RADAR
LAYER 0 FUNCTIONAL OBJECTS

Figure 11. IOM Functional Object Tree Diagram. This figure illustrates an example of an IOM Functional Object Tree Diagram from *DoD AAS ATC IOM*. This diagram illustrates the allocation of major functional objects of the *Traffic Surveillance* major functional object.

All objects below the level 1 diagram should be modeled as functional objects, where all functional objects by definition perform work, e.g., transform data and/or generate control messages based on some internal or external stimuli. Preparing *hollow bubbles* is to be avoided. Structured analysis is concerned with decomposing data transformation processes to their primitive levels by using a process of top-down function decomposition. IOMs are concerned with allocating functional objects to layers of functional capabilities and establishing information and control paths among them, to form a hierarchy of communicating functional objects. Thus, it is important to note that IOM transformation diagrams do not portray functional decomposition in the same sense as Ward-Mellor *Real-Time Structured Analysis* transformation diagrams, although some of the diagram notation is employed in a similar fashion.

7 - Model the System Aspects of the FOs at Each Diagram Level

Functional objects allocated to a particular layer or a level within a layer should be logically related and work cooperatively to accomplish the processing required by their assigned layer. System aspects that should be modeled are:

- Data flow between peer functional objects with the same parent
- Data flow between parent and child functional objects
- Data stores employed by functional objects
- Control flow between peer functional objects
- Control flow between parent and child functional objects
- Functional object behavior.

Data flows, data stores and their associated data structures are decomposed and described.

Operations that a functional object performs are explained in the textual description of the functional object. However, it would be reasonable to prepare transformatic n diagrams to describe the operations of a functional object, as shown in Figure 12 on page 31 if there is sufficient time to do so. These supplementary diagrams would be viewed as extensions to the IOM and not part of the core document.

The behavior of functional objects or their model of control can be either described in text or represented via a state transition diagram or another suitable graphical or formal notation mechanism. An example state transition diagram is illustrated in Figure 13 on page 32.

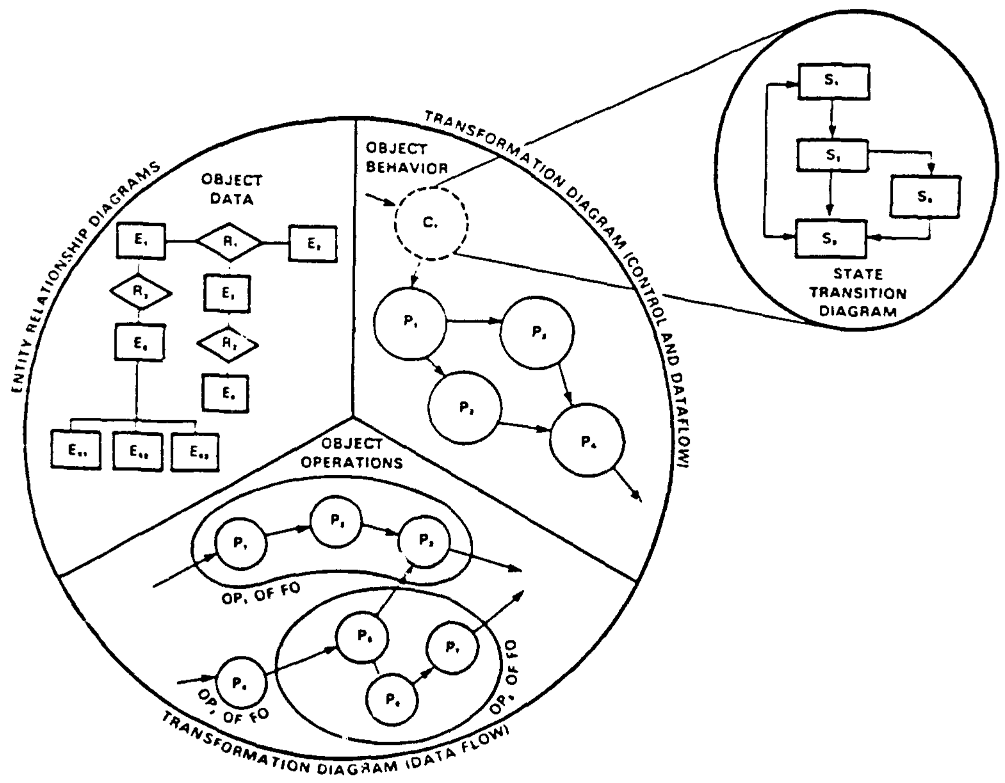

ENTITY RELATIONSHIP DIAGRAMS
OBJECT DATA
OBJECT BEHAVIOR
TRANSFORMATION DIAGRAM (CONTROL AND DATAFLOW)
OBJECT OPERATIONS
OP, OF FO
OP, OF FO
OP, OF FO
TRANSFORMATION DIAGRAM (DATA FLOW)
STATE TRANSITION DIAGRAM

Figure 12. Describing System Aspects of a Functional Object. This figure illustrates a possible extension to the IOM to characterize the system aspects of an IOM functional object.

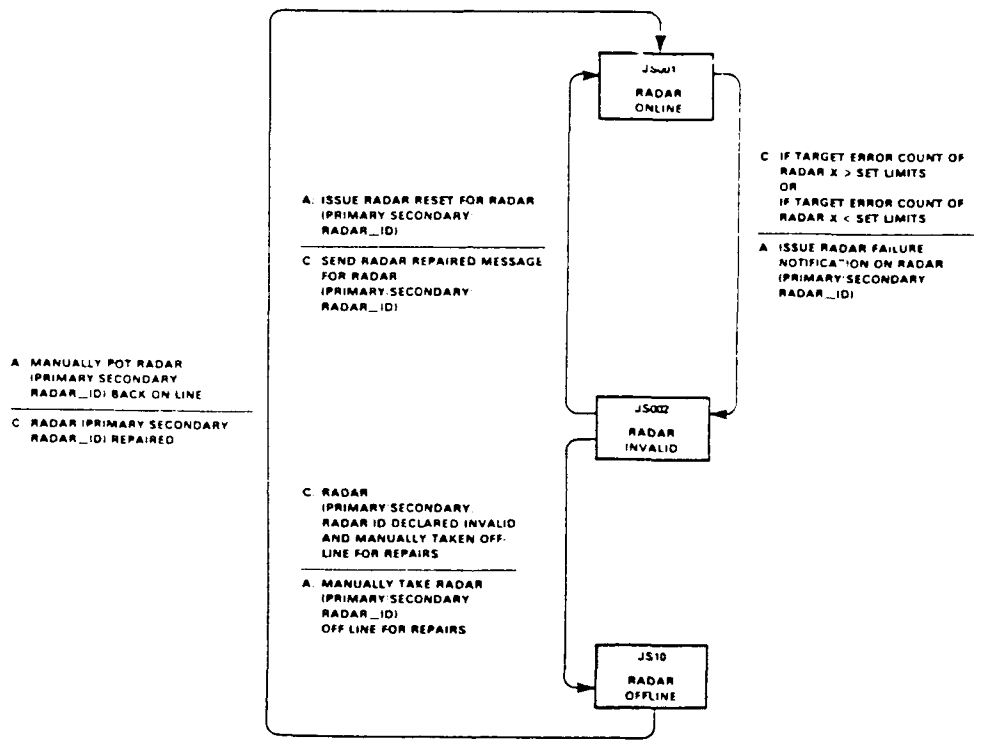

JS001
RADAR ONLINE
JS002
RADAR INVALID
JS10
RADAR OFFLINE
C IF TARGET ERROR COUNT OF RADAR X > SET LIMITS OR IF TARGET ERROR COUNT OF RADAR X < SET LIMITS
A ISSUE RADAR FAILURE NOTIFICATION ON RADAR (PRIMARY SECONDARY RADAR_ID)
A: ISSUE RADAR RESET FOR RADAR (PRIMARY SECONDARY RADAR_ID)
C SEND RADAR REPAIRED MESSAGE FOR RADAR (PRIMARY SECONDARY RADAR_ID)
A MANUALLY POT RADAR (PRIMARY SECONDARY RADAR_ID) BACK ON LINE
C RADAR (PRIMARY SECONDARY RADAR_ID) REPAIRED
C RADAR (PRIMARY SECONDARY RADAR ID DECLARED INVALID AND MANUALLY TAKEN OFF-LINE FOR REPAIRS
A. MANUALLY TAKE RADAR (PRIMARY SECONDARY RADAR_ID) OFF LINE FOR REPAIRS

Figure 13. Describing Functional Object Behavior. This figure illustrates a state transition diagram for an IOM functional object. This state transition diagram describes the behavior of 1.1 RADAR SUPERVISOR' from the *DoD AAS ATC IOM*.

8 - Model Message Communication of FOs between IOM Diagram Levels

Message communication flows should be modeled to show communication between functional objects between IOM diagram levels, namely:

- Data flow between parent and child functional objects
- Control flow between parent and child functional objects.

Data Message Communication of FOs between IOM Diagram Levels: Identify data flow/message communications between all functional objects on a given diagram, their parent functional object and their subordinate (children) functional objects, as illustrated in Figure 14 on page 34. This figure illustrates FO 7.1.1.1 TEMPERATURE "A" DIGITIZER and FO 7.1.1.2 TEMPERATURE "B" DIGITIZER sending data flows TEMP "A" and TEMP "B" to FO 7.1.1 PRODUCTION RATE CONTROL. It also shows FO 7.1.1 PRODUCTION RATE CONTROL sending data flows TEMPERATURE "A" ADJUST COMMANDS and TEMPERATURE "B" ADJUST COMMANDS to FO 7.1.1.3 TEMPERATURE "A" CONTROLLER and and FO 7.1.1.4 TEMPERATURE "B" CONTROLLER.

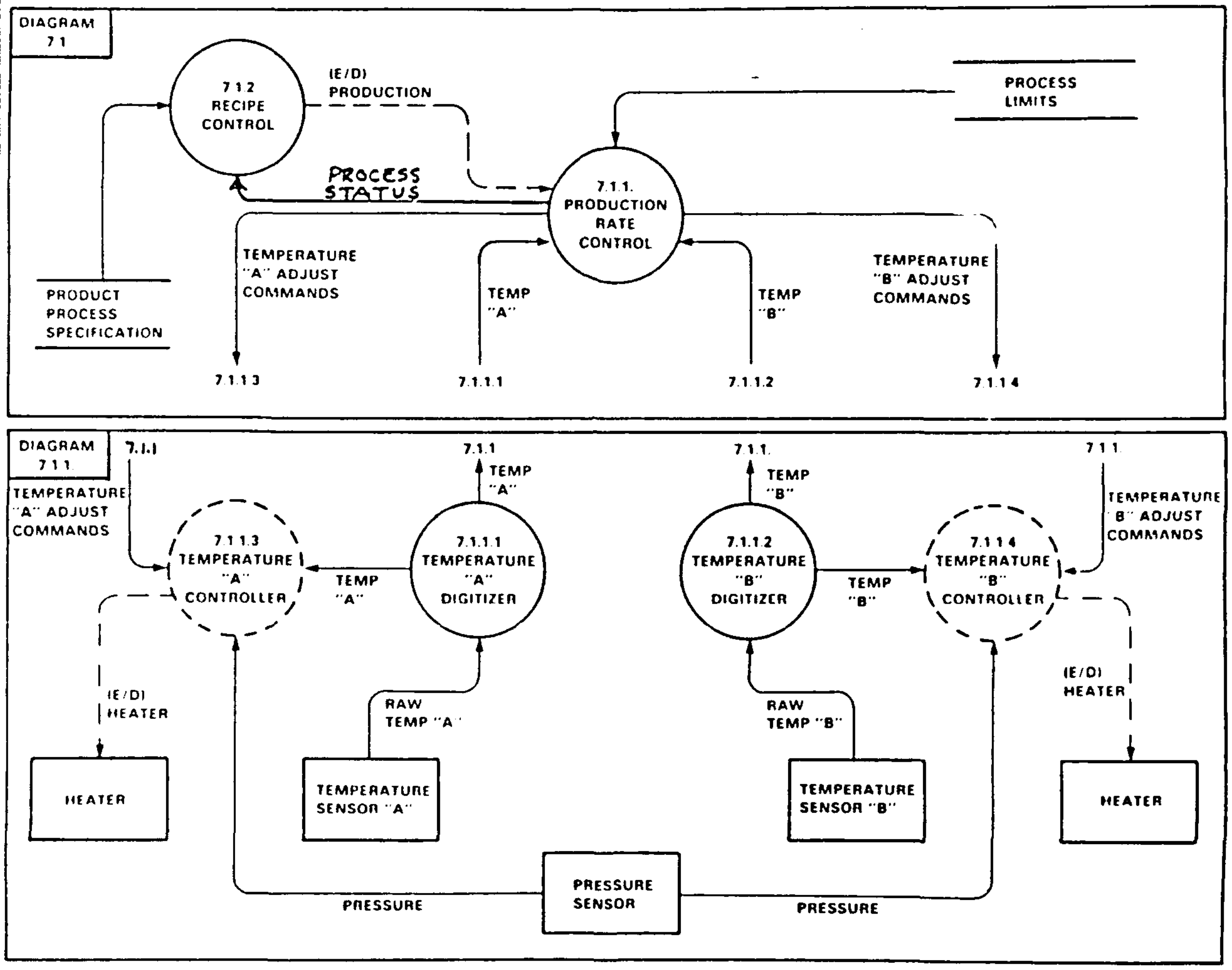


DIAGRAM 7.1
7.1.2 RECIPE CONTROL
(E/D) PRODUCTION
PROCESS LIMITS
PROCESS STATUS
7.1.1. PRODUCTION RATE CONTROL
TEMPERATURE "A" ADJUST COMMANDS
PRODUCT PROCESS SPECIFICATION
TEMP "A"
TEMP "B"
TEMPERATURE "B" ADJUST COMMANDS
7.1.1.3
7.1.1.1
7.1.1.2
7.1.1.4
DIAGRAM 7.1.1
7.1.1
TEMPERATURE "A" ADJUST COMMANDS
7.1.1.3 TEMPERATURE "A" CONTROLLER
7.1.1.1 TEMPERATURE "A" DIGITIZER
7.1.1.2 TEMPERATURE "B" DIGITIZER
7.1.1.4 TEMPERATURE "B" CONTROLLER
TEMPERATURE "B" ADJUST COMMANDS
(E/D) HEATER
RAW TEMP "A"
RAW TEMP "B"
HEATER
TEMPERATURE SENSOR "A"
TEMPERATURE SENSOR "B"
PRESSURE SENSOR
PRESSURE

Figure 14. IOM Data Flow Between Levels. This figure illustrates a finctional object sending a data flow messages from subordinate objects to a parent object, and from a parent object to subordinate objects.

Control Message Communication of FOs between IOM Diagram Levels: Identify control messages between all functional objects on a given diagram, their parent functional object and their subordinate (children) functional objects, as shown in Figure 15 on page 36. In this figure FO 6.4 HANDOFF-IN PROCESSING sends control signal messages with trigger FO 6.4.1, FO 6.4.2 and FO 6.4.3 depending on the data received.

It should be noted that there are two forms of control messages:

- Control signal - indicated by a dotted line

  Control signals can represent:

  - Triggering an operation
  - Enabling or disabling an operation
  - Suspending or resuming an operation
  - An event which causes a transition between states.

- Discrete data flow - indicated by a solid line with a double arrow.

  A discrete data flow is data that is only available at certain times, dictated by the policy of the sender. Discrete data can be used as a form of control, as the operations of a functional object will respond to the arrival of some stimuli (data message).

Notation for control flows is discussed in Appendix B of this report.

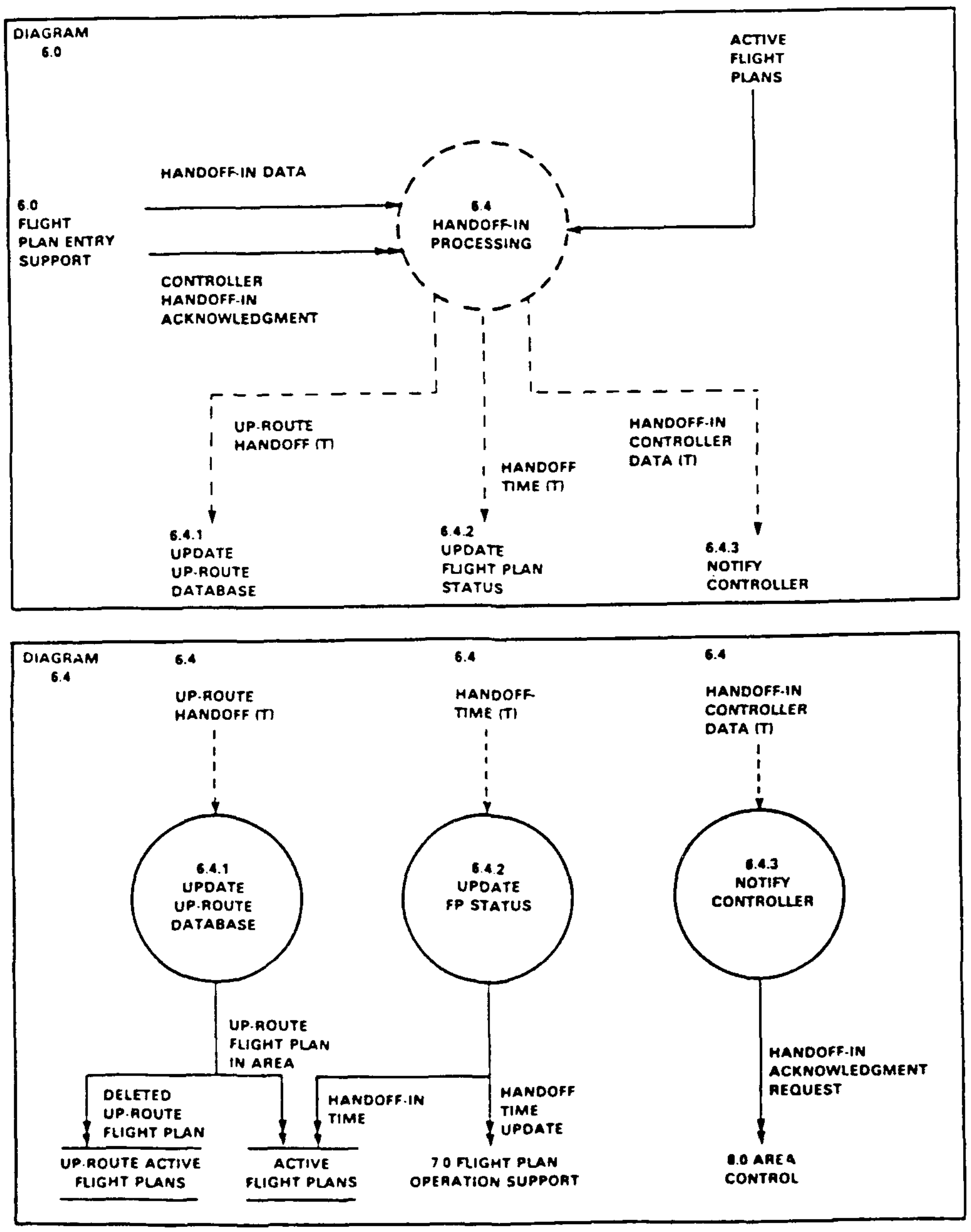


DIAGRAM 6.0
ACTIVE FLIGHT PLANS
HANDOFF-IN DATA
6.0 FLIGHT PLAN ENTRY SUPPORT
6.4 HANDOFF-IN PROCESSING
CONTROLLER HANDOFF-IN ACKNOWLEDGMENT
UP-ROUTE HANDOFF (T)
HANDOFF TIME (T)
HANDOFF-IN CONTROLLER DATA (T)
6.4.1 UPDATE UP-ROUTE DATABASE
6.4.2 UPDATE FLIGHT PLAN STATUS
6.4.3 NOTIFY CONTROLLER
DIAGRAM 6.4
6.4 UP-ROUTE HANDOFF (T)
6.4 HANDOFF TIME (T)
6.4 HANDOFF-IN CONTROLLER DATA (T)
6.4.1 UPDATE UP-ROUTE DATABASE
6.4.2 UPDATE FP STATUS
6.4.3 NOTIFY CONTROLLER
UP-ROUTE FLIGHT PLAN IN AREA
DELETED UP-ROUTE FLIGHT PLAN
HANDOFF-IN TIME
HANDOFF TIME UPDATE
HANDOFF-IN ACKNOWLEDGMENT REQUEST
UP-ROUTE ACTIVE FLIGHT PLANS
ACTIVE FLIGHT PLANS
7.0 FLIGHT PLAN OPERATION SUPPORT
8.0 AREA CONTROL

Figure 15. IOM Control Flow Between Diagram Levels. This diagram illustrates a functional object sending a control message to a functional object on its child diagram.

9 - Continue Diagram Decomposition by Using Steps 6 through 8

Continue steps 6 through 8 until layer 0 functional objects are identified. Layer 0 devices, such as sensor data acquisition devices and actuators, are shown on the lowest level diagram in the diagram decomposition.

- A separate diagram is not prepared for the Layer 0 devices; they are shown as sources in the diagram and are represented by a box, which indicates an external interface, as shown in Figure 6 on page 15. Each layer 0 device should have a general representative box in the context level diagram. For example, in the Air Traffic Control IOM, on the context level diagram we identified an interface named AIR SURVEILLANCE (see Figure 7 on page 21); at the lowest level diagram, we showed layer 0 interfaces named PRIMARY RADAR and SECONDARY RADAR (see Figure 11 on page 29).

The basic modeling process runs typically 90 to 120 days. Time should be extended for unfamiliar domains, where no formal or informal domain models exist, to permit a cursory domain analysis. Depending on the scope and complexity of the system, add 30 to 60 days to perform a cursory domain analysis.

Review and Refinement

After the basic modeling process has been completed, the first pass Information Object Model (or first pass "book") is prepared. The IOM is then presented to experts, personnel involved in the current system and management. The model is then critiqued, questions and issues to be addressed are submitted and considered, and the IOM is updated accordingly. The first pass book is to be prepared within the first 45 to 60 days of the specification modeling effort, depending upon the size and complexity of the job The second pass book is subsequently prepared and presented. Usually only two complete Information Object Models are prepared. However, a systems engineer/analyst will produce many versions of the model of the major functional objects to which he or she is assigned. These models will be prepared from interviews with system and domain experts and will be reviewed with them, and validated as much as possible, before the systems engineer/analyst writes his or her IOM sections.

The Review and Acceptance Phase

After the final IOM has been completed, a formal review meeting is held with the customer. The purpose of this meeting is to explain the IOM. After this review has been completed, the customer and contractor decide whether to proceed into the implementation modeling phase or a rapid prototype development phase. Even if the customer does not wish to proceed after the IOM is completed, he has a technology-independent logical design of a proposed system that describes the system's essential processing requirements. Thus, the IOM could be used as the basis for procuring all or part of the proposed system at a later date, when the company is ready to proceed.

Cost Estimation

The completed IOM can be used as the basis for estimating the cost of a proposed system. As the IOM is an organization of hierarchically distributed objects, costs could be estimated for integrating or modifying existing off-the-shelf "objects" or building non-existing "objects." Using this approach, DoD program managers could determine where best to allocate resources to build a needed system, given a multi-year procurement process.

Packaging the Information Object Model

The IOM is packaged to provide differing levels of information to satisfy the information needs of various levels of corporate personnel. Figure 16 on page 40 illustrates this packaging concept. A typical IOM should contain the following material:

1. Introduction

   The introduction should contain relevant information about the task and relevant problem domain information, as well as important issues to be resolved. It should also include any recommendations for further analysis to be performed.

2. Mission Statement

   This is the mission statement prepared and accepted during the planning phase. It provides the statement of need the system is to satisfy, the scope of what the system is to address and a brief operational concept of the system (graphically depicted, if possible).

3. The System Overview

   The system overview describes the overall system and the major functional objects (functional objects identified on the first-level diagram). The system overview must contain the following graphics and an explanation of them:

   a. System context diagram (level 0 diagram)

   b. First-level diagram (level 1 diagram)

4. The Major Functional Object Models

   The major functional object models provide a detailed description of aspects of the major functional object. The major functional object should consist of the following:

   a. The major functional object "View-From" diagram and discussion

   b. The major functional object "Interfaces" diagram and discussion, which illustrates external interfaces identified in the context diagram and internal interfaces identified in the "View-From" diagram

   c. A list of the inputs and outputs of the major functional object and their definition

   d. The major functional object "Functional Object Tree (FOT)," which illustrates the hierarchic organization of the subordinate functional objects and their communication paths; the FOT also serves as an IOM compliance tool, to ensure that modeling guidelines have been followed

   e. A discussion of the subordinate diagrams, which describe the processing of the functional object. This should include:

      1) A level I + 1 diagram and its description

      2) A description of the functional objects in the diagram

      3) A description of the inputs and outputs for each functional object on the diagram

      4) A graphic of a functional object's state transition diagram, where required

      5) The process descriptions of the functional object provided in pseudo-English or an appropriate PDL

   f. The summary descriptions from system aspects that appear on the diagrams that describe the major functional object:

      1) Information flows for the major functional object illustrating the inputs to and outputs from each of the functional objects on the subordinate diagrams describing the major functional object (Excelerator *Transformation Graph Analysis Report*)

2) All data flows
3) All data stores
4) All records and their elements
5) All elements
6) All control flows
7) All control transforms.

5. The Appendix

The appendix provides relevant supplementary data and reports as deemed necessary. Typically a report describing globally defined data stores is provided in the appendix.

The *STARS Structured Specification for the DoD Advanced Automation System* (also referred to as the *DoD AAS IOM*) produced under the IBM STARS contract provides an example of this IOM packaging concept.

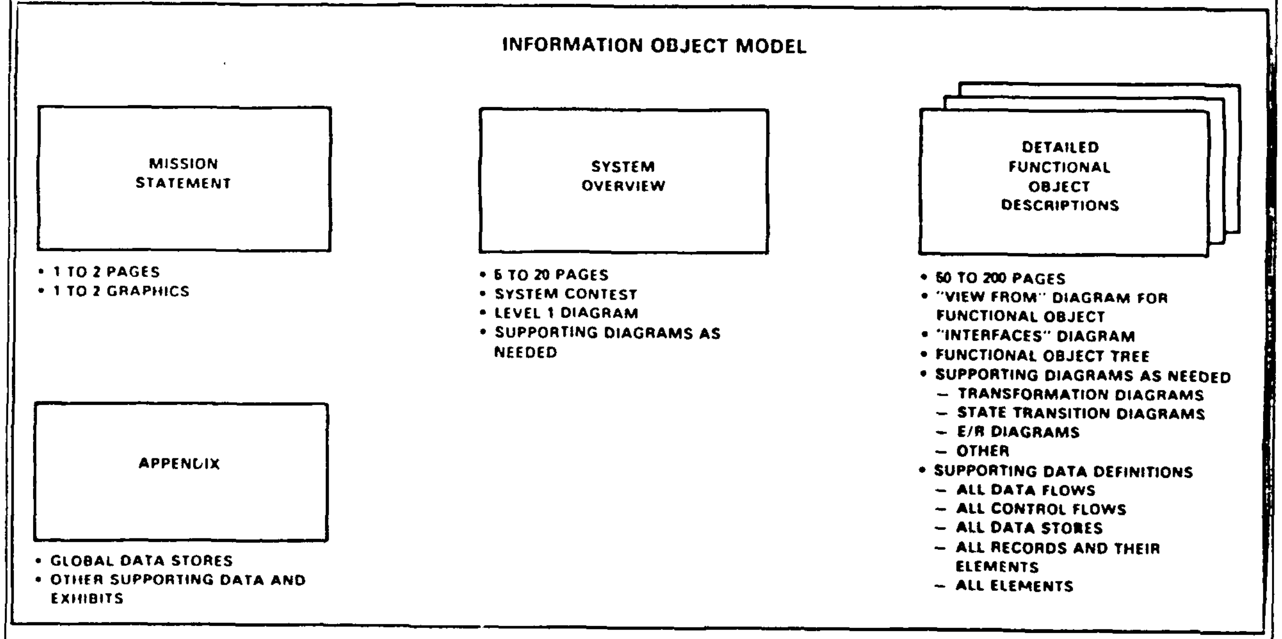


Figure 16. IOM Packaging Concept. This figure illustrates an IOM packaging of IOM components, including the mission statement, overview, detailed functional object descriptions, etc.

Guidance for Preparation and Review of IOM Diagrams

Although Information Object Model diagrams look very similar to those of Ward-Mellor *Real-Time Structured Analysis* transformation diagrams, they differ in several respects. It is these differences that separate Information Object Modeling from structured analysis, namely:

1. All "functional objects" portrayed on an IOM transformation diagram must themselves perform work; they must not just represent an encapsulation of lower level functionality.
2. A "functional objects" portrayed on an IOM transformation diagram should pass data and/or control messages to at least two of the three of the following, as illustrated in Figure 17 on page 42:
   - A peer functional object of the same parent
   - Its parent functional objcct
   - Its child functional object.

   It should be recognized that this requirement does not apply to layer 0 devices, as they represent the devices that interface with the real-world objects. Thus, layer 0 devices are drawn on the IOM transformation diagrams by using a rectangle symbol to identify an interface.

This section will illustrate several modeling problems that the IBM team encountered during the building of our *DoD AAS ATC* IOM. It should be recognized that the modeling rules presented in this report, apply only to preparing IOM transformation diagrams that follow the *IOM functional object allocation approach to modeling*, as described in previous sections. Four types of problems will be discussed:

- Confusion between IOM layers and modeling levels
- Functional object communication problems
- Non-communication peer functional objects
- Existence of *hollow bubbles*.

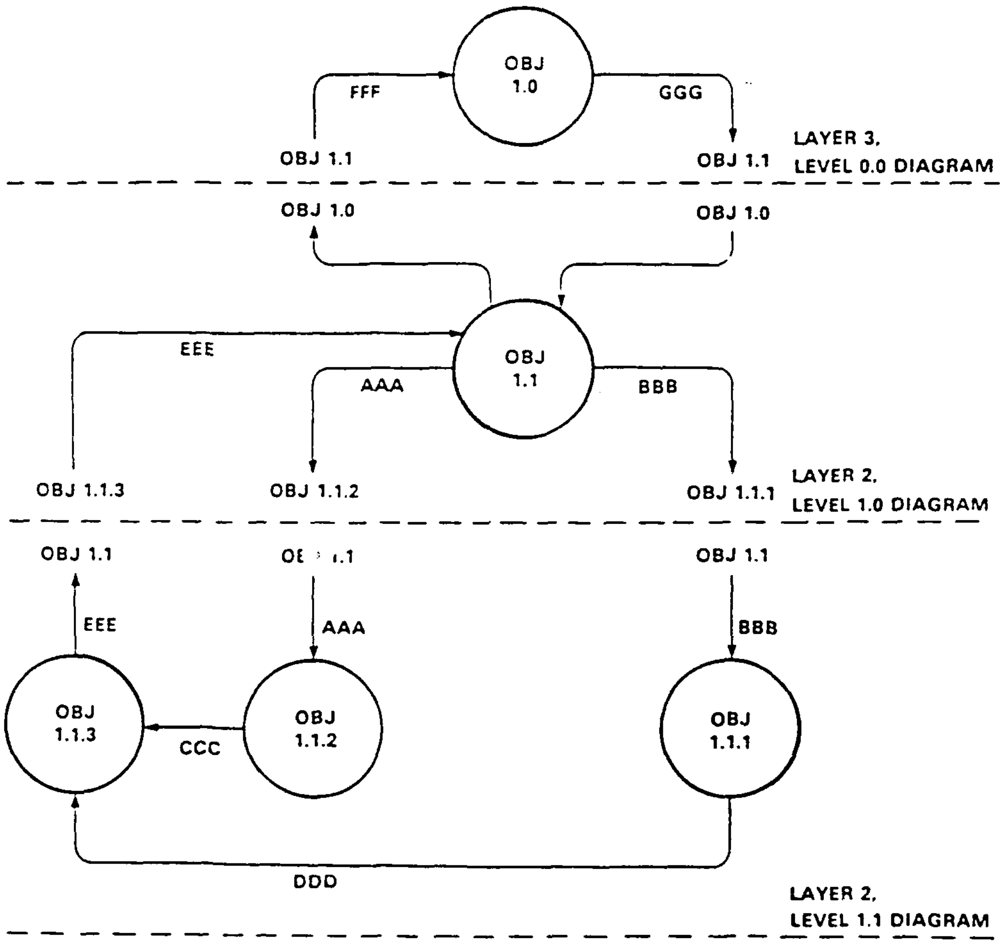


Figure 17. IOM Object Communication: Parent, Child, Peer. This figure illustrates that all functional objects should have at least two of the three forms of communication: 1) communication (data control) between peer objects with the same parent, 2) communication (data control) between a functional object and its sibling objects, and 3) communication (data control) between a functional object and its parent object.

Confusion Between IOM Layers and Modeling Levels

Capabilities are assigned to each functional object, on the basis of the layered model. Further, it is possible for multiple levels to exist for an IOM layer, as illustrated in Figure 18 on page 44. This figure illustrates that an IOM layer can be comprised of multiple levels, where a diagram is prepared for each level. Note that layer 2 requires three diagram levels to model, as FO 2.5, FO 2.5.1 and FO 2.5.1.6 are all layer 2 functional objects.

Diagrams are prepared for each level identified, until a layer 0 device is allocated. Sensors and actuators are the lowest level devices to be depicted in an IOM. This concept is illustrated in Figure 6 on page 15, where temperature sensors and heaters are shown on the same level with the layer 1 devices to which they report and by which they are controlled.

Errors can be introduced when the concept of layers versus levels is not clearly understood and properly applied. It is permissible to use as many levels as necessary to model an IOM layer. However, if the number of levels exceeds 5, a problem may exist.

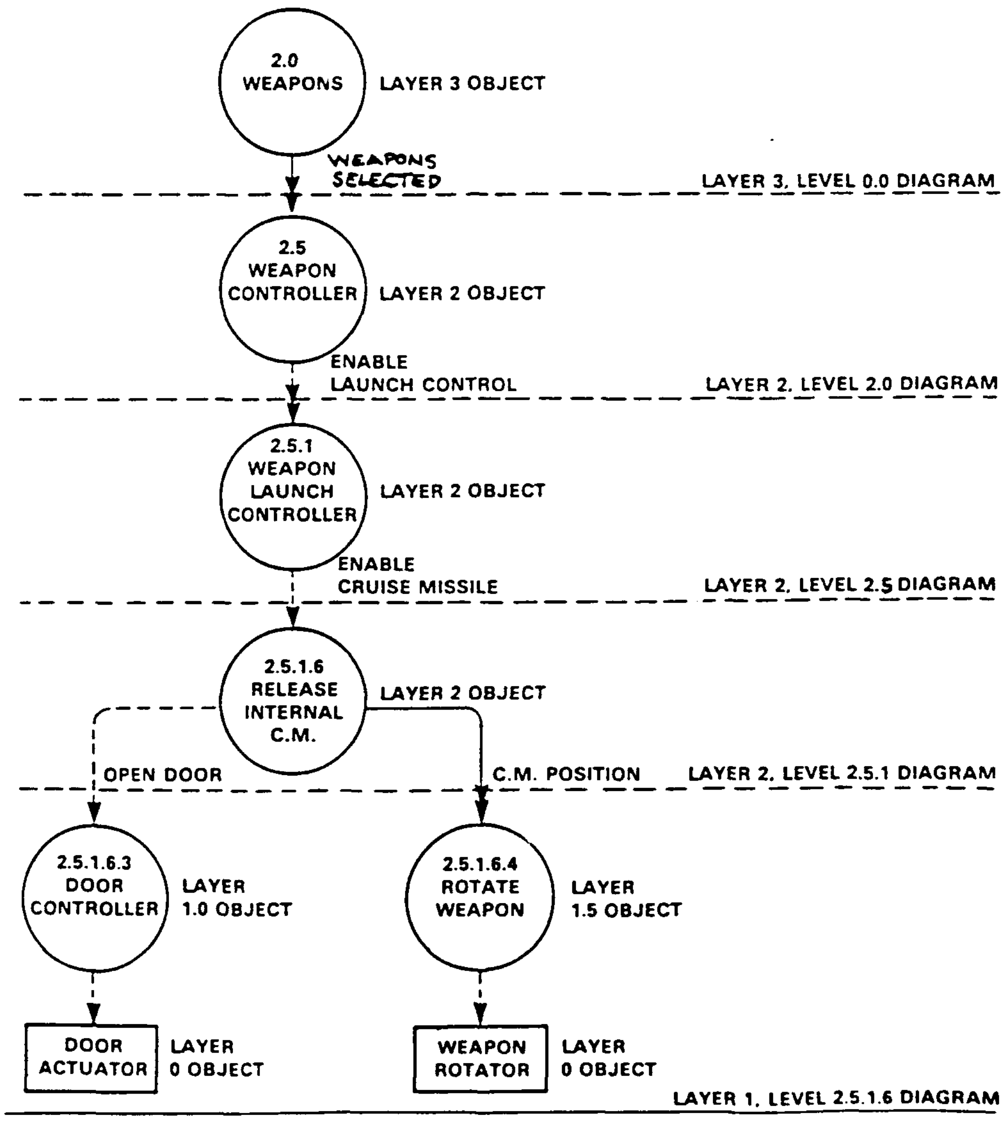


Figure 18. IOM Layers versus Levels. This figure illustrates an IOM hierarchical thread from the 2.0 Weapons Functional Object, which shows communication between IOM layers and levels. Also note that an IOM layer can comprise several levels.

IOM Diagram Problem: Functional Object Communication Problems

A functional object should communicate with its parent object, peer objects sharing the same parent object or its subordinate (children) objects. Because the IOM represents a model of hierarchically organized functional objects, if the above communication rules are rigorously applied, functional object communication problems should not exist.

Two problems illustrated in Figure 19 on page 46 are:

- If an object bypasses a level, communicating directly with a grandparent object, a modeling problem exists
- If an object communicates with a cousin object, a modeling problem exists.

Potential causes of the problems include:

- The use of "Hollow Bubbles"
- Improper leveling in the diagrams
- Improper functional object allocation.

Problems may be identified by reviewing the functional object tree to examine its placement in the object hierarchy.

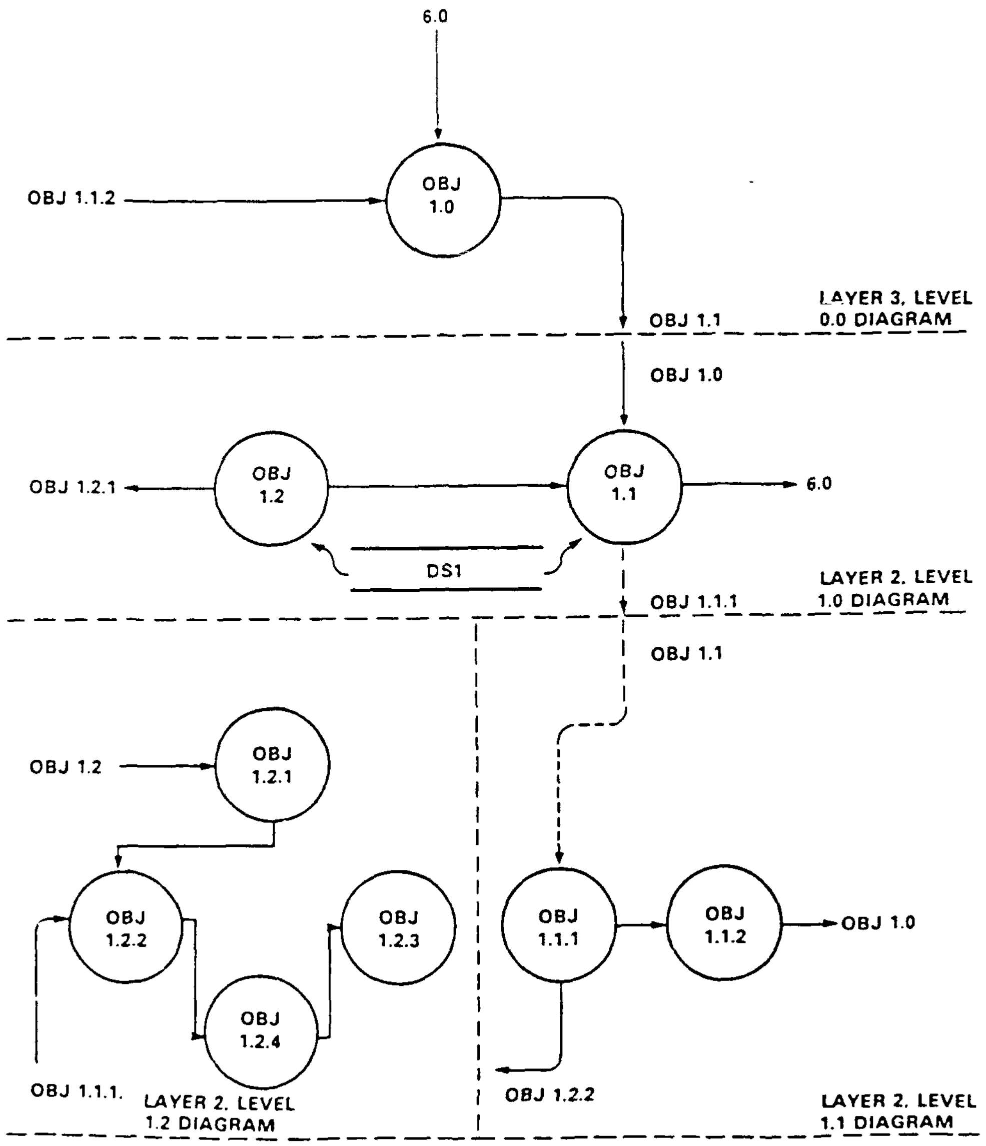


Figure 19. Possible IOM Functional Object Communication Problems. This diagram illustrates two possible communication problems, namely:

- Typical objects communicate with their peers, or with parent or child levels. There may be a problem with object 1.1.2, which directly sends data to object 1.0, and thus bypass its parent diagram object
- Object 1.1.1 communicates with object 1.2.2: in an IOM, cousin objects should not communicate.

IOM Diagram Problem: Non-Communicating Peer Functional Objects

Functional objects at any diagram level should have some form of communication with peer, parent or children objects. When non-communicating peer functional objects are discovered, an examination of the functional objects should be made, as illustrated in Figure 20 on page 48.

Potential causes for the problems include:

- Improper allocation of functional objects; objects may need to be promoted to another level
- An error in the diagram.

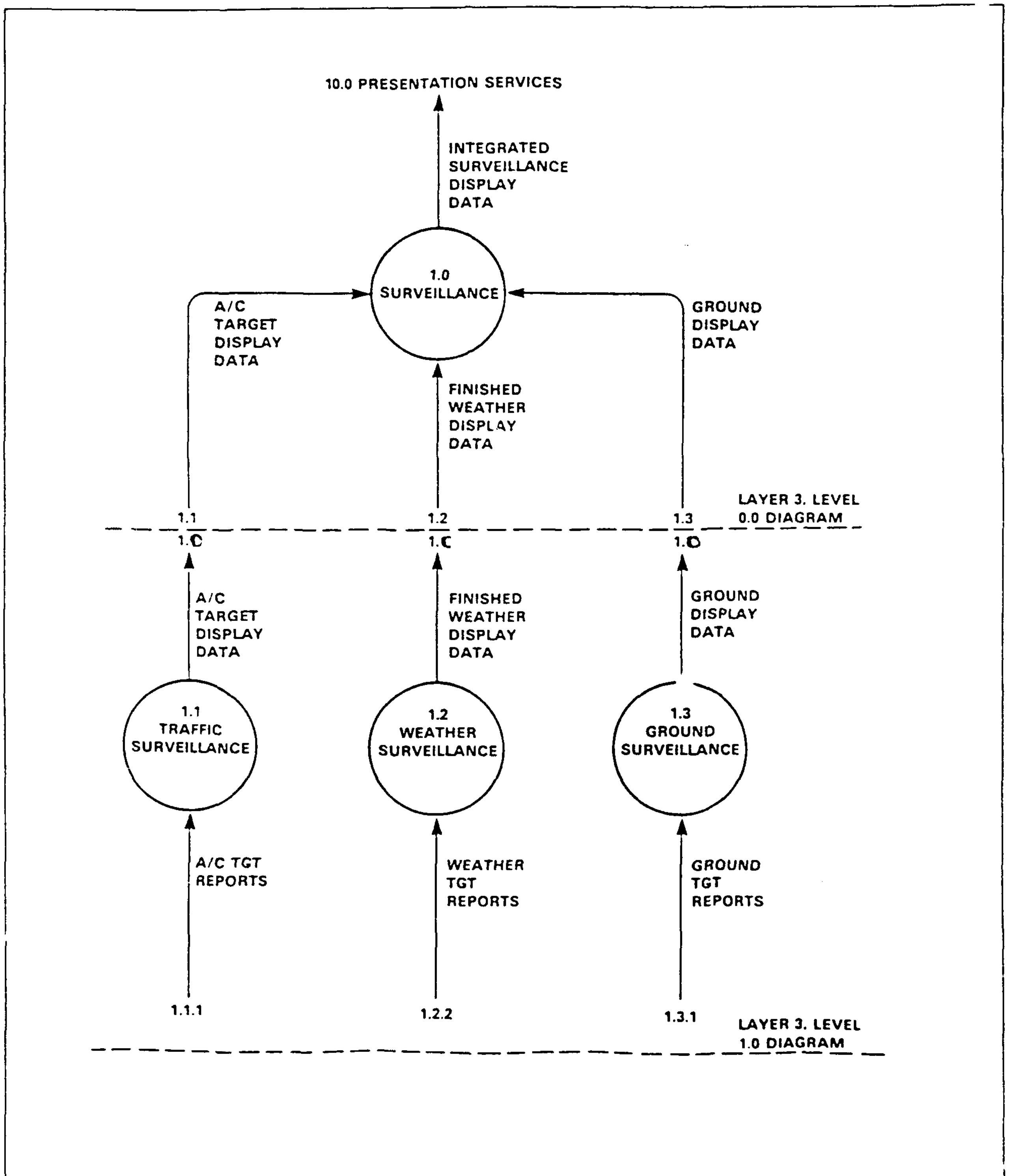


Figure 20. Non-Communication Peer Functional Objects. This figure represents a perfectly reasonable IOM drawing, where functional objects illustrate communication between child and parent drawing objects. However, 1.1, 1.2 and 1.3 do not communicate with each other. There is no peer to peer data or control communication. 1.1, 1.2 and 1.3 may need to be promoted or coalesced.

IOM Diagram Problem: Existence of "Hollow Bubbles"

In performing Information Object Modeling, it is important to remember that all functional objects should perform work. Although there may be cases where this is awkward, the allocation of functional objects should be re-examined to see whether the awkwardness can be corrected.

Hollow bubble is a term that is used to describe the packaging of seemingly related processes for the convenience of process abstraction and encapsulation. The technique of process abstraction is used heavily in structured analysis.

Hollow bubbles can usually be spotted when a functional object:

- Violates the functional object communication rules
- Appears to be a message pass-through.

Note that in Figure 21 on page 50 "1.0 PROCESS SURVEILLANCE DATA" is a hollow bubble, which encapsulates the functional objects:

- 1.1 PROCESS TRAFFIC SURVEILLANCE DATA
- 1.2 PROCESS WEATHER SURVEILLANCE DATA
- 1.3 PROCESS GROUND SURVEILLANCE DATA.

The packaging on 1.0 PROCESS SURVEILLANCE DATA hides the fact that traffic, weather and ground surveillance should be promoted, because they each own their own radars and share information with each other. Both the *Traffic Surveillance* and the *Weather Surveillance* functional objects were promoted in the *DoD AAS Area Control Facility* IOM, as shown in Figure 8 on page 23. *Ground Surveillance* was allocated to the *DoD AAS Tower Control Facility* IOM.

Also note that the three data flows illustrated entering functional objects 1.1, 1.2 and 1.3 respectively are presented on the structured analysis diagram for the sake of "data flow" accounting, even though the data is employed at lower levels. This is typically not the case with an IOM representation. Data is processed by functional objects that receive it, as data flow is presented within the context of the functional object hierarchy. Thus, flows do not need to be carried from level to level until they are employed.

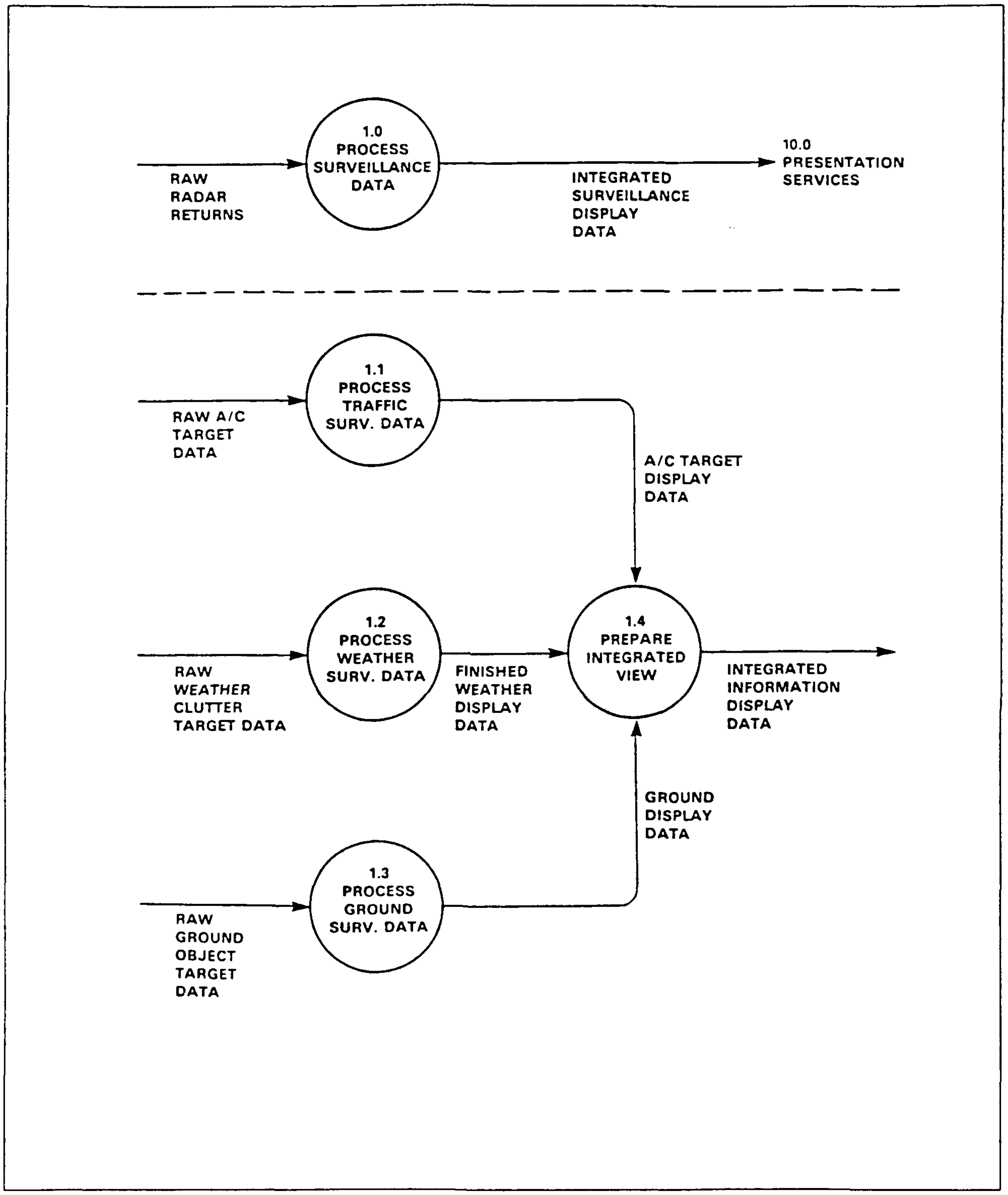


Figure 21. "Hollow Bubbles" Portraying Functional Objects. This diagram illustrates 1.0 as a "hollow bubble." 1.0 represents a packaging of processes for convenience and may obscure the fact that a reorganization of functional objects may be desirable. Also note that this diagram, which represents a structured analysis view of "1.0 PROCESS SURVEILLANCE DATA," does not highlight message communication as an IOM representation would.

Issues Raised from Our Experience with Information Object Modeling

The purpose of this section is to address several issues that surfaced during the course of STARS task QM15. The central issues that will be addressed are:

- Is the Information Modeling methodology, based on the *IOM functional object allocation approach* and the *White Layered Model*, a good general purpose modeling technique for developing system specification models?
- What is the role of the IOM in the Software-First Life Cycle methodology?
- What is role of the IOM in the DoD systems procurement process?

We shall briefly discuss these issues in this section, and shall then offer conclusions drawn from our experience on STARS Task IQM15.

The Layered Model as a General Purpose Model for System Specification

The Information Object Modeling methodology was developed from Dr. White's experiences with industrial real-time process control and Computer Integrated Manufacturing systems. The modeling methodology is predicated on the idea that an analyst can identify and allocate functional objects within the framework of the "layered model." The layered model, as presented in a previous section, represents a reasonable way to model the allocation of objects and their communication paths for systems that exhibit distributed hierarchic control.

Is the "Layered Model" Extendable?

If it is reasonable to assume that all systems could be represented by a set of hierarchically organized abstract machines, then the "layered model" paradigm could be extended to specify information systems, that exhibit little in the way of distributed control. Rather, information systems exhibit threads of processing where data is manipulated and stored in a database and is extracted and further processed to present information in the form of screen displays or reports.

A thesis that remains to be explored in the future is wheth :here are analogous "layered models for different classes of systems similar to the one described in this report for the *distributed hierarchic control layered model* (also known as the *White Layered Model*). Further, would these "layered models" (to be used to allocate functional objects and to establish communication paths between them) be useful to specify systems via establishing machine-independent logical designs for them? From increasing our knowledge about specific problem domains and identifying and building models of objects, we might be able to design systems based on hierarchically organizing objects associated with their "domain-specific layered model of capabilities." In other words, we would like to find "layered models" similar to the *distributed hierarchic control layered model* for different problem domains and to be able to apply the modeling techniques for these domain-specific layered models in a similar fashion.

When Foxboro applies these techniques, they apply them to specify and design real-time distributed process control systems. They understand the object types that they will use in building most process control systems. Once they understand the process to be controlled, they can organize their objects into a processable paper model by using the *distributed hierarchic control layered model.* The keys to their success are their domain models of process control and their domain models of manufacturing processes for which they have developed process control systems. These models enable them to view seemingly unrelated processes, such as the manufacturing of chocolate and iron ore, and to identify the similarities between aspects of the two processes. Thus, they serve as a means to reuse parts of existing models. For example, Dr. White once observed that there were similarities between the manufacturing processes of both chocolate and iron ore, in that the quality of both products was measured by the granularity of the raw material employed, e.g., the best quality chocolate depended on the how fine the cocoa beans were ground, etc.

Without good domain models it is difficult to identify where a process paradigm from one domain might fit another. This is the most important lesson that we learned from the QM15 experience. We need to develop domain models to support the efficient specification and design of systems by using the domain models to identify and reuse objects with common characteristics and operations.

The Role of the IOM in the Software-First Life Cycle Methodology

In viewing the concept of a reuse-driven life cycle that employs concurrent software engineering development practices, it is clear from experience that only systems-in-the-small are amenable to rapid prototyping from scratch or from using component reuse libraries, given the current state-of-the-practice. For prototyping and developing systems-in-the-large, we must have some understanding of the problem to be solved, before we can develop a prototyping plan. Further, we must have some understanding of the application domain, as the problem domain may constrain how a system is to be designed. The Information Object Model can serve to provide a basic understanding of system requirements and a logical framework of objects that need to be identified in existing component repositories and can be reused or modified or be built from scratch. In this sense, IOM modeling can serve as a candidate methodology for performing a *Preliminary Systems Analysis* as defined in the IBM QM15 STARS CDRL 1240, *Software-First Life Cycle Final Definition* . The major activities of the Software-First Life Cycle are illustrated in Figure 22 on page 53.

It should be recognized that as we improve our ability to employ a reuse-driven life cycle for the development of software, the role that the IOM could play in the Software-First Life Cycle may differ significantly.

```
    | Desired Operational
    |    Capability
   \|/
+--------+               +---------+              +----------+  +---------+
|Prelim. |  ...........  | System  |  ..........  | Product- |  |System   |
|System  |  .Pre-     .  | Archi-  |  .PreProd-.  | ization  |  |Operation|
|Analysis|->.Prototyp-.->| tecture |->.uctiz-  .->|    &     |->| and     |
|        |  .ing      .  |         |  .ation   .  |Production|  |Support  |
|        |  .Review   .  |         |  .Review  .  |          |  |         |
+--------+  ...........  +-+---+---+  ..........  +----------+  +----+----+
/|\ /|\ |                  |   |  /|\                      /|\  /|\  |
 |   |  |                  |   |   |                        |    |   |
 |   |  |  New Technology  |   |   |                        |    |   |
 |   |  |     Requests     |   |   +------------+           |    |   |
 |   |  +----------------->|   |                |           |    |   |
 |   |                     |   |                |           |    |   |
 |  \|/                    |   |               \|/         \|/  \|/  |
 |  /----------\ . . . . . . . . . . . . .  /----------\ . . . . . . |
 |  |Repository|           |   |            |Repository|             |
 |  |          |           |   |            |          |             |
 |  +----------+ . . . . . . . . . . . . .  +----------+ . . . . . . |
 |      |                  |   |               /|\                   |
 |      |          New     |   Component        |                    |
 |      |          Technology  Development      |                    |
 |      |          Requests|   Requests         |                    |
 |      |                 \|/ \|/               |                    |
 |      |                 +---------+           |                    |
 |      +---------------->|         |<----------+                    |
 |                        | Software|                                |
 |                        | Growing |                                |
 |                        |         |                                |
 |                        +---------+                                |
 |                                        New Operational Capability |
 +-------------------------------------------------------------------+
```

Figure 22. Software-First Life Cycle - Conceptual View. This figure shows the major activities of the Software-First Life Cycle.

The Reuse-Driven Software-First Life Cycle

IBM STARS views systems development from two different perspectives:

- The reusable products system development life cycle
- The systems development life cycle, which employs reuse engineering products.

The first perspective addresses the development of reusable products from a domain analysis of a particular problem domain by identifying common problems and developing solutions to solve those common problems. The second perspective addresses the use of reusable products in developing systems. From our experiences with Foxboro, we recognized that both perspectives of systems development are needed.

One of the reasons Foxboro can effectively compete in integrating industrial process control systems is that they possess models of the process control problem domain and the hardware and software components required to develop solutions. The IOM, viewed in this context, seems very practical as it represents an allocation of functional objects, where each functional object has a real-world set of components that can be used to instantiate the functional objects in the IOM. Foxboro can achieve this because of their expertise of the process control system problem domain. This is why IOMs are much easier for Foxboro to build and why IOMs serve as excellent vehicles for precedented systems. There are challenges associated with each systems integration effort, as unique requirements may be presented. However, most of the systems Foxboro develops have some common process control paradigms from which Foxboro can draw, in the specification modeling process. Foxboro's knowledge of process control and its paradigms for process control system development constitute Foxboro's reuse engineering products. The development of an IOM represents the application of Foxboro's reuse engineering products.

Building IOMs without a good understanding of the problem domain may in fact be more difficult than performing a structured analysis, because the functional objects required to prepare the IOM may be extremely hard to recognize. For example, if one does not know what a sensor is, what its attributes are, what its interfaces are, and what functions it performs, one will have a hard time recognizing it as a functional object.

It is important to recognize that the successful development of an IOM for an unprecedented system or an unfamiliar application domain requires that a domain analysis be performed to identify the common objects of that domain and to identify:

- Their attributes
- Their interfaces
- The operations they perform
- Their behavior, given certain stimuli.

As mentioned earlier, one technique to accomplish this is to build a generic IOM for the system. However, it must be recognized that the generic IOM may require 45 to 60 days, in addition to the 90 to 120 days for the problem-specific IOM.

The Role of the IOM in the Spiral Model of Software Development

The first set of activities addressed in the Spiral Model (Boe-01) of the software process for each spiral cycle includes: 1) establishing the objectives for a portion of a product to be developed (mission statement), 2) examining various candidate methods for implementing and identifying the constraints to be imposed on candidate methods of implementation (trade studies, based on identified functional objects), and 3) identifying risks associated with each candidate implementation method. Given that an IOM could be incrementally refined to support the specification portion of the activities identified above, one could prepare partial or complete IOMs, addressing only those major functional objects considered candidates for development in a given cycle of the spiral. For example, the mission statement, system context, level 1 diagram, and the "View-From" diagrams could be prepared for only those functional objects targeted for prototype development. This would provide the necessary scoping on the analysis and modeling that would be performed. The resulting models could be used for assessing candidate implementation methods and could serve as a model for allocating risk items as well as estimating costs. This would be particularly effective, if real-world instances of each functional object could be identified as part of the spiral implementation study.

The Role of the IOM in the DoD Systems Procurement Process

Given the length of current DoD procurement cycles, devoting five to seven months for the development of an Information Object Model for proposed complex systems seems to be a good idea. The Information Object Model could be used to describe the major functional objects of a system in sufficient detail to use as a cost estimation tool and could be used as a tool to plan for procuring a complex systems-in-the-large weapons system. On the basis of our experiences, given the right system, meeting these goals seems achievable. I feel that the five to seven month figure is realistic. This time period would cover:

- Analysis effort planning
- Team orientation
- Methodology and tools training

- A cursory problem domain analysis
- Problem-specific IOM development.

The DoD program manager for a new DoD systems-in-the-large system could commission a multi-disciplined, multi-organizational team to perform a problem domain analysis and to develop an IOM. The DoD program manager could use this IOM to gain a better understanding of:

- The system he needs to procure and what might be involved to build it
- What technology breakthrough challenges need to be solved
- How system procurement might be approached.

The IOM could be used as a vehicle for system procurement. The IOM could be used as a specification for a contractor prototype competition, where individual contractors would be asked to prototype one or more major functional objects. Because the IOM minimizes functional cohesion and is based on the principle of hierarchically organized communicating objects, the IOM could potentially be used as a technique for compartmentalizing classified system procurements.

Much more work would be required to substantiate all of the above ideas.

Conclusions

The QM15 experience was an extremely valuable one for the STARS program. Most of its value was in the thinking the three STARS prime contractors put into the process of how to initiate the Software-First Life Cycle and to recognize the importance of domain analysis as a precursor activity to systems analysis. Even more important was the validation of the concept of the twin systems development life cycles, namely:

- The reusable products system development life cycle
- The systems development life cycle, which employs reuse engineering products.

Intuitively this appeared to be a good idea; however, through our involvement with Dr. Gerald White of Foxboro we know that it works for a specific problem domain.

One can show that the IOM modeling methodology provides techniques to produce a more terse model than its real-time systems specification methodology counterparts. However, other real-time system specification methodologies such as Ward-Mellor, Pirbai-Hatley, etc., approach the description of the processing a system performs in a more thorough manner. After identifying the functional objects and modeling them, analysts typically describe the operations they perform with text. This approach to describing system functionality is more ad hoc in the Information Object Modeling methodology than in its real-time systems specification methodology counterparts.

The Information Object Modeling methodology is not a general purpose technique and is most useful for modeling systems that exhibit natural hierarchic control. From our experience with the Information Object Modeling methodology and our interactions with Dr. White of Foxboro, I believe the strengths of the Information Object Modeling methodology are that it:

- Partially supports object-orientation, where structured analysis techniques do not
- Fosters a more terse model than yielded by a real-time structured analysis.

However, during the application of the Information Object Modeling methodology, we were reminded of a few axioms that make any analysis effort successful. These include recognizing:

- That the analysis effort is more efficient when the analysis team is given time to properly understand the target problem domain
- That customer and contractor management buy-in and support are important

- That analysis is more effective when modeling is not based on documentation alone - talking to people improves the understanding process.

Finally, the Information Object Modeling methodology requires analysts to possess expert interviewing skills. During both phase I and phase II, we practiced structured interviewing techniques similar to those employed by knowledge engineers. Notes extracted from the May 1990 *STARS Monthly Status Report* on interviewing techniques are included in appendix C of this report.

The Information Object Modeling methodology is useful for modeling real-time systems that exhibit hierarchic control. Where a problem domain exhibits characteristics that fit the hierarchic control paradigm, the Information Object Modeling methodology is an effective technique to employ in conducting a *Software-First Preliminary Systems Analysis* or for supporting the objectives task and implementation candidate identification and assessment activities of the *Spiral Model of Software Development and Enhancement*. However, on the basis of the IBM STARS QM15 experience, the IOM modeling techniques were more difficult to apply where there was no model of natural hierarchic control, as expressed by the *White Layered Model*. The *DoD AAS Area Control Facility* IOM, with few exceptions, spanned only layers 3 and 2. The only two functional objects that possessed level 0 devices were FO 1.0 TRAFFIC SURVEILLANCE and FO 2.0 WEATHER SURVEILLANCE. However, the radars that provided surveillance data were not controlled by a parent functional object. The processing of the radar data was a data flow process. The DoD AAS Area Control Facility exhibited little in the way of control, as it is an information processing system. The aspects of control, as in the control of aircraft, are outside the boundaries of the system itself. From this experience, IBM learned that the Information Object Modeling methodology is a hard technique to apply to developing information processing systems, which do not exhibit the properties of hierarchic control.

Given that a set of layers could be formulated for systems with data flow as their processing threads, similar to that of the layered model, the IOM methodology would be more powerful as a general systems development methodology. However, a layered model for a data-driven information processing system probably does not exist. I base this conclusion on the fact that Dr. White uses structured analysis techniques to model systems within layers 4 and 5 of his model, as evidenced by his work in the CIM Reference Model (Wil-01). However, I cannot say that a layered model for data-driven information systems does not exist. However, any future efforts applied towards IOMs should investigate identifying alternative layering schemes by analyzing selected problem domains. From an analysis of the objects of a selected problem domain, a set of layers may be identified and refined that are analogous to the *White Layered Model*, and an attempt could be made to use this layered model to develop an IOM based on the methodology described in this report.

Appendix A. References

1. Blu-01 - *Executing Real-Time Structured Analysis Specifications* by R. Blumofe and A. Hecht; From *Software Engineering Notes*, Volume 13, Number 3, July 1988.
2. Boe-01 - *A Spiral Model of Software Development and Enhancement* by Barry W. Boehm; from*IEEE Computer*, May 1988.
3. Dem-01 - *Structured Analysis and Systems Specification* by Tom DeMarco; Published by the Yourdon Press, 1978.
4. Wil-01 - *A Reference Model for Computer Integrated Manufacturing* edited by Theodore J. Williams; Published by Instrument Society of America, 1989.
5. War-01 - *Structured Development for Real-Time Systems*, Volumes 1 through 3, by Paul T. Ward and Stephen J. Mellor; Published by the Yourdon Press, 1986.
6. Whi-01 - *Real-Time Management ... A Second Industrial Revolution* by Gerald R. White; from the *Proceedings of the Purdue ALC Conference*, September 1987.

Appendix B. IOM Diagram Notation

Information Object Modeling employs most of the basic graphical tools employed and described in the textbook *Structured Systems Development for Real-Time Systems*, Volume 1 (War-01). This section will describe the basic graphical tools used in an IOM and discuss the notation and symbology used for the IOM transformation diagram.

IOM Transformation Diagram

Many examples of IOM transformation diagrams have been illustrated throughout this report. A transformation diagram shows an arrangement of functional objects, data stores, and control transforms that are connected as required by data and control flows. Table 1 provides a summary of the types of flows used in preparing transformation diagrams. Table 2 on page 61 provide: a summary of the symbols used in preparing transformation diagrams.

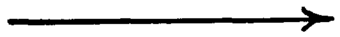
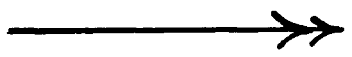
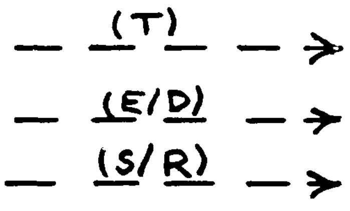
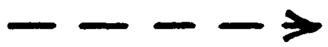
Transformation Diagra Flows		
Symbol Type	Description	Notation
Continuous Data Flow	Represents data that is continuously available to all functional objects that employ it	
Discrete Data Flow	Represents data that is available, based on the policy of its provider. A discrete data flow can serve as a mechanism to trigger an operation within a functional object.	
Control Flow	Represents a "data-less" prompt or trigger that initiates a functional object to perform specified operations. The five most common control flow types are: (T) trigger an operation, (E/D) enable/disable an operation, (S/R) suspend/resume an operation.	
Event Flow	Represents a "data-less" signal what represents an event that will result in a transition between states.	

Table 1. Flows Employed in Transformation Diagrams. This table provides an enumeration and definition of flow types used on an IOM transformation diagram.

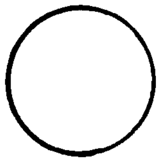
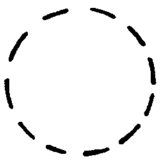
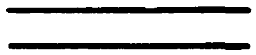
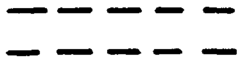
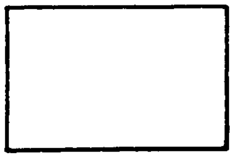
Transformation Diagram Symbols		
Symbol Type	Description	Notation
Functional Object	An object that performs work and has the following characteristics: performs operations, based on message stimuli (receipt of data and/or control flows) and exhibits behavior (based on the processing it performs, can effect its own state changes). A functional object is the basic unit of allocation in an IOM.	
Control Transform	A state machine used to determine the sequencing of events and initiation of operations of one or more functional objects.	
Data Store	Represents a repository of persistently available data.	
Control Store	Represents a repository of event flow states. (This is rarely employed in an IOM.)	
Sensor/Actuator (Layer 0 Device)/External Interfaces	Represents an object, external to the system that communicates between the originator (source) or receiver (sink) of data and interface to the functional object represented in the IOM. Typically these interfaces are layer 0 devices. However, external interfaces may communicate at any layer appropriate within an IOM, if a higher layer device directly employs or provides information to an interface.	

Table 2. Symbols Employed in Transformation Diagrams. This table provides an enumeration and definition of the symbol types used on an IOM transformation diagram.

Entity-Relationship Diagrams

Entity-relationship diagrams may be used as an optional IOM exhibit to illustrate entities and their relationships to each other. These diagrams may also be employed to describe data internal to each functional object. Guidelines for preparing Entity-Relationship diagrams and a discussion of the symbology and notation for preparing them can be found in the textbook *Structured Development of Real-Time Systems*, Volume 1 (War-01).

State-Transition Diagrams

State-Transition diagrams illustrate the interior of a control transform that illustrates the states the control transform can have, the transition conditions required to change from one state to another and the transition outputs that occur, when the transition input conditions have been met. An example of a State-Transition diagram is presented in Figure 13 on page 32. Guidelines for preparing State-Transition diagrams and a discussion of the symbology and notation for preparing them can be found in the textbook *Structured Development of Real-Time Systems*, Volume 1 (War-01).

Appendix C. Interviewing Techniques

The following notes were extracted from the May 1990 *QM15 STARS Monthly Status Report*, which discussed observations and lessons learned about interviewing:

- The Information Object Modeling methodology as a systems analysis exercise is an extremely useful vehicle for gaining knowledge about a complex system. The technique is based on structured interviewing using a model referred to as the "White Five Layer Model" as a guidance tool for maintaining proper levels of abstraction with an interviewee, and for using an object-oriented approach to examine objects and for performing object decomposition. Team members who knew absolutely nothing about the B1-B bomber felt that they learned a great deal in a short period of time, by using Dr. White's techniques. These techniques are very similar to knowledge engineering techniques employed to acquire knowledge from domain experts to develop expert systems.
- Expert interviewing techniques are essential, especially where there is little time to complete a project and where people resources are the key to fast and efficient data extraction for modeling. The following observations were made on interviewing domain experts:
  - Describe to the interviewee the objectives for the interview, along with your information goals
  - Use your information goals to keep the interviewee on track
  - Let the interviewee know that there are no wrong answers and that he (or she) should indicate when he knows he is correct, or when he is is guessing
  - Permit topic regression to let the interviewee get comfortable with the interview situation and to give the interviewer some understanding of his "technical" passions
  - Use the best tools for collecting the information of interest and do not hesitate to switch tools when one is not working (e.g., data flow, control flow, state transition, ERA diagrams, etc.)
  - Different techniques are required for interviewing interviewees with various levels of experience and expertise; do not stretch the limits of your interviewee.

Appendix D. STARS Task IQM15: The Information Object Modeling Methodology

This appendix contains the briefing charts for the presentation entitled *STARS Task IQM15: The Information Object Modeling Methodology* This briefing was given at the *STARS Task QM15 - SEI Software Engineering Architecture Project Technical Interchange*, held on June 18 through June 19, 1990 in Pittsburgh, Pennsylvania.

STARS Task IQM15 The Information Object Modeling Methodology Presentation Version 1.0

18 June 1989

W. H. Ett

IBM Corporation
Federal Sector Division
800 North Frederick Avenue
Gaithersburg, Maryland 20879

Unclassified

Preface

This briefing was prepared for the STARS Task QM15 - Software Engineering Institute Technical Information Exchange, held on June 18 through June 19, 1990 in Pittsburgh, Pennsylvania.

Any opinion or position expressed in this presentation are my own, and not necessarily those shared by IBM.

Contents

PURPOSE OF THIS TALK

- Discuss the motivations behind STARS Task QM15
- Introduce the Information Object Model
- Introduce the Information Object Modeling methodology
- Discuss Information Object Modeling in context the SDLC
- Discuss lessons learned and present conclusions

STARS TASK IQM15 - SOFTWARE FIRST SYSTEMS ANALYSIS

- Government-run experiment on learning and applying a commercially-developed specification modeling methodology

- Task intended to address the issue of assembling a sufficient specification for use in the Software-First Life Cycle

- Task goal of producing a system specification useful to support Ada software design

STARS TASK IQM15 - SOFTWARE FIRST SYSTEMS ANALYSIS (Cont.)

- The QM15 Experiment
  - Teach the methodology to a team composed from members of the three STARS prime contractors
  - Learn the methodology by applying it to a DoD "systems-in-the-large" problem
  - Have the three primes apply the methodology to selected DoD "systems-in-the-large" problems
    - IBM - Military Air Traffic Control / DoD AAS
    - Unisys - NCCS Afloat (C3 Battle Management)
    - Boeing - Learned and built a Foxboro-style Implementation Model for the B-1B

- Assertions to be proved
  - A team can produce a specification for a complex DoD system in 90 days
  - The IOM can be used as the entry point for "Software Growing" of the Software-First Systems Life Cycle
  - The Information Object Model techniques are "object-oriented"

MOTIVATIONS BEHIND STARS TASK QM15

- Foxboro develops complex system specifications in 90 days
- Foxboro practices reuse in process control system development
- Techniques may be of value in initiating the SFLC
- Techniques would be invaluable if a reasonable specification could be assembled in 90 days
  - As a procurement planning tool
  - A means for planning a system prototyping effort

INFORMATION OBJECT MODEL INTRODUCTION

- What is an Information Object Model?
- What is the Information Object Modeling Methodology?
- What is a Functional Object?

WHAT IS AN INFORMATION OBJECT MODEL?

- A specification model produced using the Information Object Modeling methodology

- A specification model is:
  - A model of the requirements for a proposed system
  - A machine-independent logical view prepared
    - Uses "how" tools to describe "what" the system shall do
  - A processable model that can be tested

- An information packaging concept

WHAT IS THE INFORMATION OBJECT MODELING METHODOLOGY?

- Method for rapidly accumulating and assimilating knowledge
- Method of modeling using functional objects
  - Method of organizing functional objects (FOs), based on their capabilities and roles
  - Modeling control and message paths between objects in an object hierarchy
- Method for packaging the work products of the analysis effort

WHAT IS A FUNCTIONAL OBJECT?

- The basic unit of allocation in an IOM
- An object that performs work
- An object that satisfies the following criteria:
  - Has identifiable characteristics
  - Performs one or more operations
  - Has the ability to receive from and/or send messages to other FOs
  - Has internal data
  - Has one or more potential states
  - Exhibits behavior depending on stimuli

INFORMATION OBJECT MODELING METHODOLOGY INTRODUCTION

- The Two Approaches for building Information Object Models
  - Structured Analysis Process Decomposition Approach
  - IOM Functional Object (FO) Allocation Approach

- Work products of both can be prepared using the IOM packaging concept

STRUCTURED ANALYSIS PROCESS DECOMPOSITION APPROACH

- Modeling is performed via process decomposition - White Example in CIM Reference Model
  - Context diagram is prepared (system context)
  - Level-1 diagram is prepared (major functions of system)
  - Diagrams are prepared for each process being decomposed
  - Lowest level diagram shows primitives for process

- This approach was followed in preparing the B-1B IOM, with a few exceptions:
  - Context diagram is prepared (system context - object, not process)
  - Level-1 diagram is prepared (major functional objects (MFOs) of system)
  - Real-time structured analysis used to model each MFO below level-1
  - External interfaces shown on lowest level diagrams (e.g. sensors)

IOM FUNCTIONAL OBJECT ALLOCATION APPROACH

- Modeling is performed using the IOM functional object allocation approach:
  - Context diagram is prepared (system context - object, not process)
  - Level-1 diagram is prepared (MFOs of system)
  - Lower level diagram levels hierarchy of communicating FOs
    - FOs are identified and allocated to a specific "layered model" layer
    - Processing for layer is accomplished based on FOs role
    - Multiple diagram levels may be required to describe a "layered model" layer
  - External interfaces shown at level data is consumed

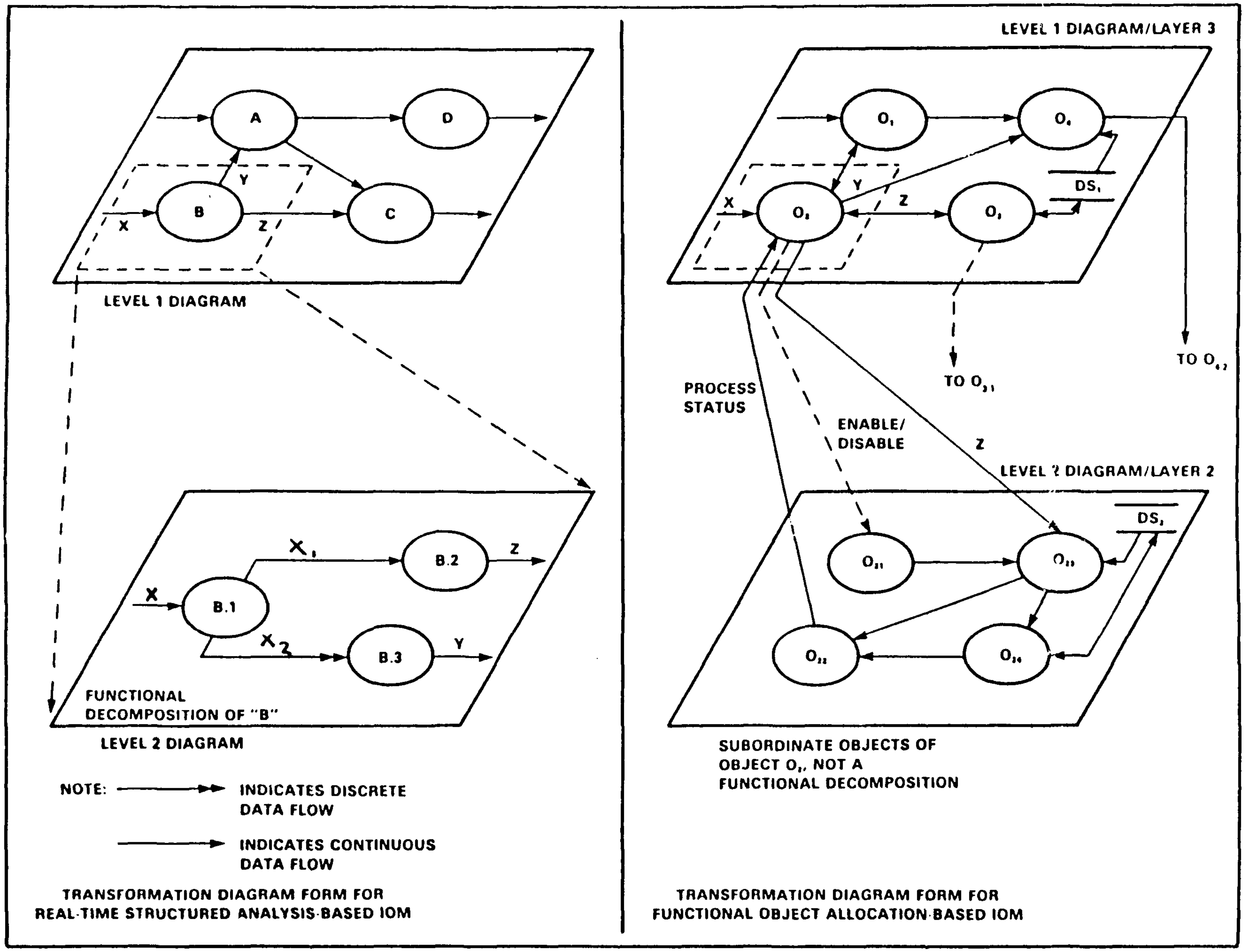
- This approach is the basis of the "Information Object Modeling" methodology

THE TWO IOM APPROACHES

See Figure Next Page

THE TWO IOM APPROACHES

Figure 1



TRANSFORMATION DIAGRAM FORM FOR REAL-TIME STRUCTURED ANALYSIS-BASED IOM

TRANSFORMATION DIAGRAM FORM FOR FUNCTIONAL OBJECT ALLOCATION-BASED IOM

THE WHITE LAYERED MODEL

LAYER 5 - STRATEGIC PLANNING (Corporate management)

LAYER 4 - MANAGEMENT CONTROL (Planning production)

LINE BETWEEN BATCH AND ONLINE TRANSACTION PROCESSING TO REAL TIME

LAYER 3 - REAL-TIME DECISION SUPPORT LAYER (Allocating and supervising materials and resources)

DIFFERENCES BETWEEN 2, 1.5 AND 1 ARE LEVELS OF DEVICE INTELLIGENCE

LAYER 2 - SUPERVISORY CONTROL LAYER (Coordinating multiple manufacturing processes and operations)

LAYER 1.5 - ADAPTIVE CONTROL LAYER (Commanding device sequences and motion of analog devices; interfaces with special purpose sensor devices)

LAYER 1 - LOGIC CONTROL LAYER (Commanding device sequences and motion of digital devices; interfaces with sensors)

LAYER 0 - SENSOR / ACTUATOR LAYER (Activates sequences and motions of devices; senses desired aspect of a manufacturing process)

LAYERS REPRESENT PROCESSING CAPABILITIES

LAYERED MODEL APPLIED TO A GENERAL PROCESS CONTROL APPLICATION

```
                              RTM     (1)                        LAYER III
                               |
      +--------------+---------+-------+--------------+-------  ...
      |              |                 |              |
     DCS            DCS               DCS            DCS (2-20)  LAYER II
      |              |                 |              |
      .              |                 .              .
      .              |                 .              .
                     |
      +----------+---+-----+----------+ ...
      |          |         |          |
     PLC        PLC       PLC        PLC  (50 - 100)             LAYER I
      |          |         |          |
      .          .         |          .
      .          .         |          .
                           |
            +----------+---+-----+----------+ ...
            |          |         |          |
         ACTUATOR   SENSOR     ACTUATOR   SENSOR  (30 - 50)      LAYER 0
```

RTM=Real Time Manager
DCS=Distributed Control System
and PLC=Programmable Logic Controller

MAPPING OF LAYERED MODEL LAYERS TO B-1B WEAPONS SYSTEM LAYERS

o LAYER 3, REAL-TIME MONITORING SYSTEM LAYER	-- WEAPONS CONTROL
o LAYER 2, DIGITAL CONTROL SYSTEM LAYER	-- LAUNCH CONTROL
o LAYER 1.5, CONTROLLER LAYER	-- LAUNCH SEQUENCER
o LAYER 1, PROGRAMMABLE LOGIC CONTROLLER LAYER	-- DOOR CONTROL
o LAYER 0, SENSOR LAYER	-- WEAPON SYSTEM SENSORS

MAPPING OF LAYERED MODEL LAYERS TO B-1B WEAPONS SYSTEM LAYERS

```
                       Weapons                          (LAYER 3)
                       Control
                          |
... ----------------------+----------------------------- ...
                          |
                       Launch                           (LAYER 2)
                       Control
                          |
... ----------------------+----------------------------- ...
                          |
                       Launch                         (LAYER 1.5)
                       Sequencer
                          |
... ----------------------+-------------------+--------- ...
                          |                   |
                       Door                Bomb         (LAYER 1)
                       Control             Release
                          |                Control
                          |                   |
... -----+----------------+--... ...-+--------+--------- ...
         |                |          |        |
       Door            Door         Bomb     Bomb        (LAYER 0)
       Actuator        Sensor       Release  Rack
                                    Actuator Sensors
```

IOM FORM EXAMPLE

See Figure Next Page

IOM FORM EXAMPLE

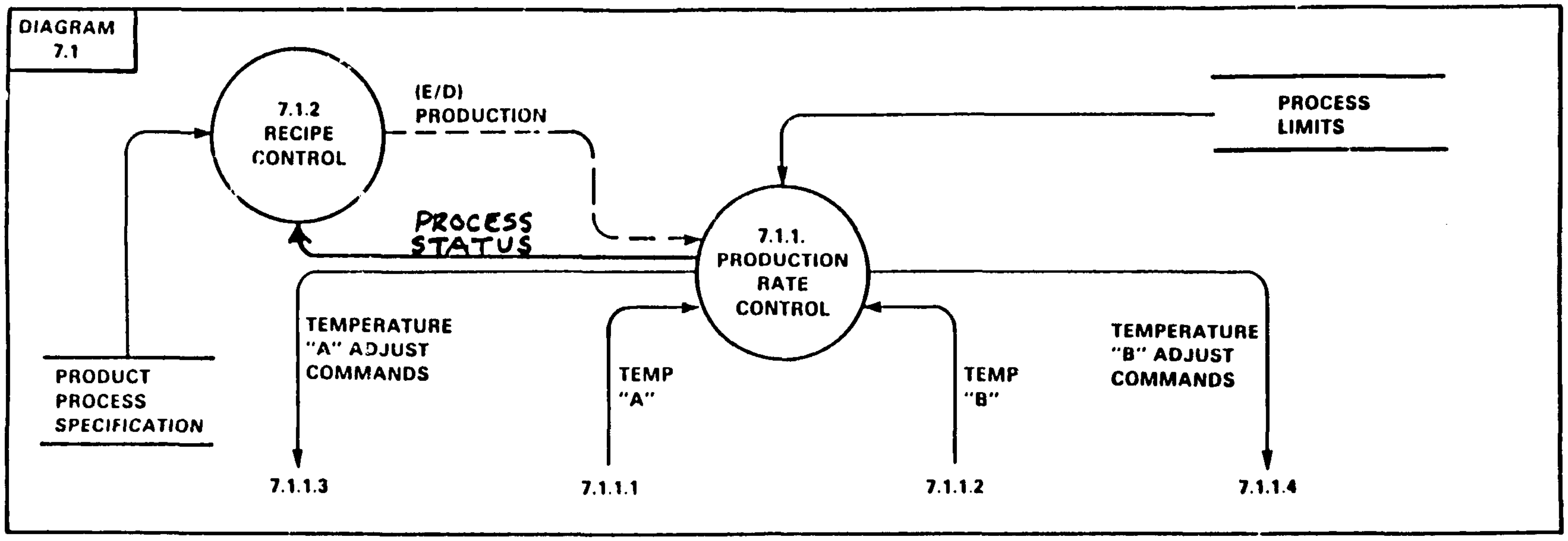
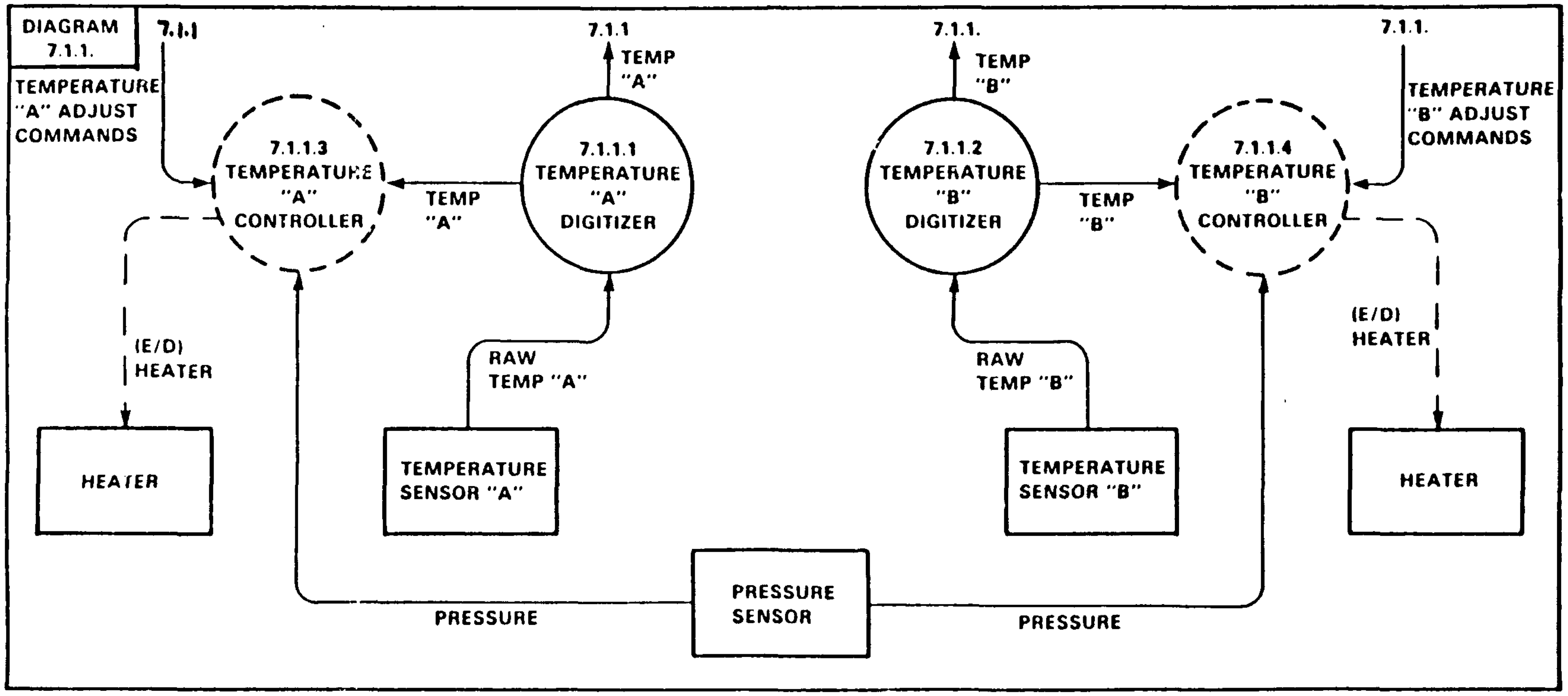


DIAGRAM 7.1
7.1.2 RECIPE CONTROL
(E/D) PRODUCTION
PROCESS LIMITS
PROCESS STATUS
7.1.1. PRODUCTION RATE CONTROL
PRODUCT PROCESS SPECIFICATION
TEMPERATURE "A" ADJUST COMMANDS
TEMP "A"
TEMP "B"
TEMPERATURE "B" ADJUST COMMANDS
7.1.1.3
7.1.1.1
7.1.1.2
7.1.1.4



PROCESS DECOMPOSITION OF FO 7.1.1 PRODUCTION RATE CONTROL

```
7.1.1  PRODUCTION RATE CONTROL
       7.1.1.1  INITIATE REFINING PROCESS
       7.1.1.2  ACQUIRE SET POINTS
       7.1.1.3  MONITOR TEMPERATURE
                7.1.1.3.1  DIGITIZE TEMPERATURE
                7.1.1.3.2  CHECK TEMPERATURE AGAINST SET POINTS
       7.1.1.4  ADJUST TEMPERATURE
       7.1.1.5  TERMINATE REFINING PROCESS.
```

BUILDING THE INFORMATION OBJECT MODEL

- Phases of building the IOM:
  - Planning phase
  - Information Object Modeling phase
  - Review, Refinement and Acceptance

PLANNING THE INFORMATION OBJECT MODELING EFFORT

- Planning phase steps:
  - Obtain management commitment
  - Prepare mission statement
  - Select the IOM analysis team
  - Establish project standards and conventions
  - Identify activities and prepare IOM development schedule
  - Collect available documentation
  - Identify domain and system experts
  - Schedule initial target interviews
  - Conduct analysis team kickoff

THE INFORMATION OBJECT MODELING PHASE

- Information Object Modeling steps:
  - Build a generic domain-specific IOM (optional)
  - Build a documentation roadmap
  - Build the first-pass application specific IOM
  - Build the second-pass application-specific IOM

THE GENERIC IOM

- Addresses system from a generic view
- Provides an overview of a complex system
- Identifies MFOs and their characteristics
- Illustrates relationships between the MFOs
- Provides a vehicle to understand the application domain
- Takes 45 - 60 days in addition to the 90 day IOM

THE BASIC MODELING PROCESS

- Establish the system context for the IOM (level-0 diagram)
- Identify the MFOs for the system (level-1 diagram)
- Identify information pipes between the MFOs (level-1 diagram)
- Identify commonly-shared data sources (level-1 diagram)
- Prepare IOM overview
- Identify and allocate subordinate FOs to each MFO
- Model the system aspects of the FOs at each diagram level
- Model message communications of FOs between IOM layers
- Continue diagram decomposition until layer 0 devices are modeled
- Incrementally validate and refine as necessary

ESTABLISH SYSTEM CONTEXT FOR THE IOM

See diagram on next page

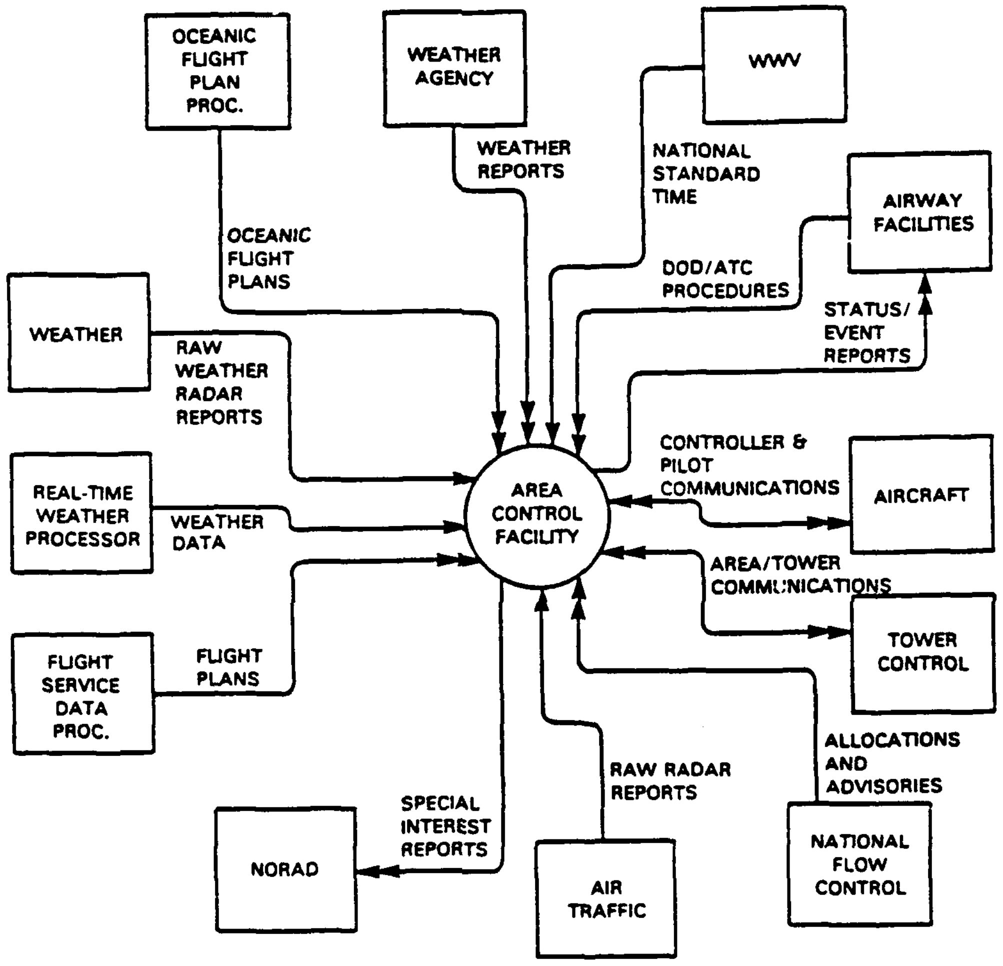


Figure 9. DoD AAS Area Control System Context Diagram. This figure illustrates the external factors which the Area Control System must interface. The diagram also establishes the system boundary for the Area Control System and illustrates the information flow between external interfaces.

IDENTIFY THE MAJOR FUNCTIONAL OBJECTS

- Shown on the level-1 diagram
- Layer 3 Functional Objects
- Must perform work
- Communicates with and controls subordinate objects

IDENTIFY INFORMATION PIPES BETWEEN THE MFOs

- Information pipes are drawn to show the message path between FOs
- "View-From" diagram technique is used to focus on a particular FO
- "Interfaces" diagram is used to show all FO external and internal interfaces

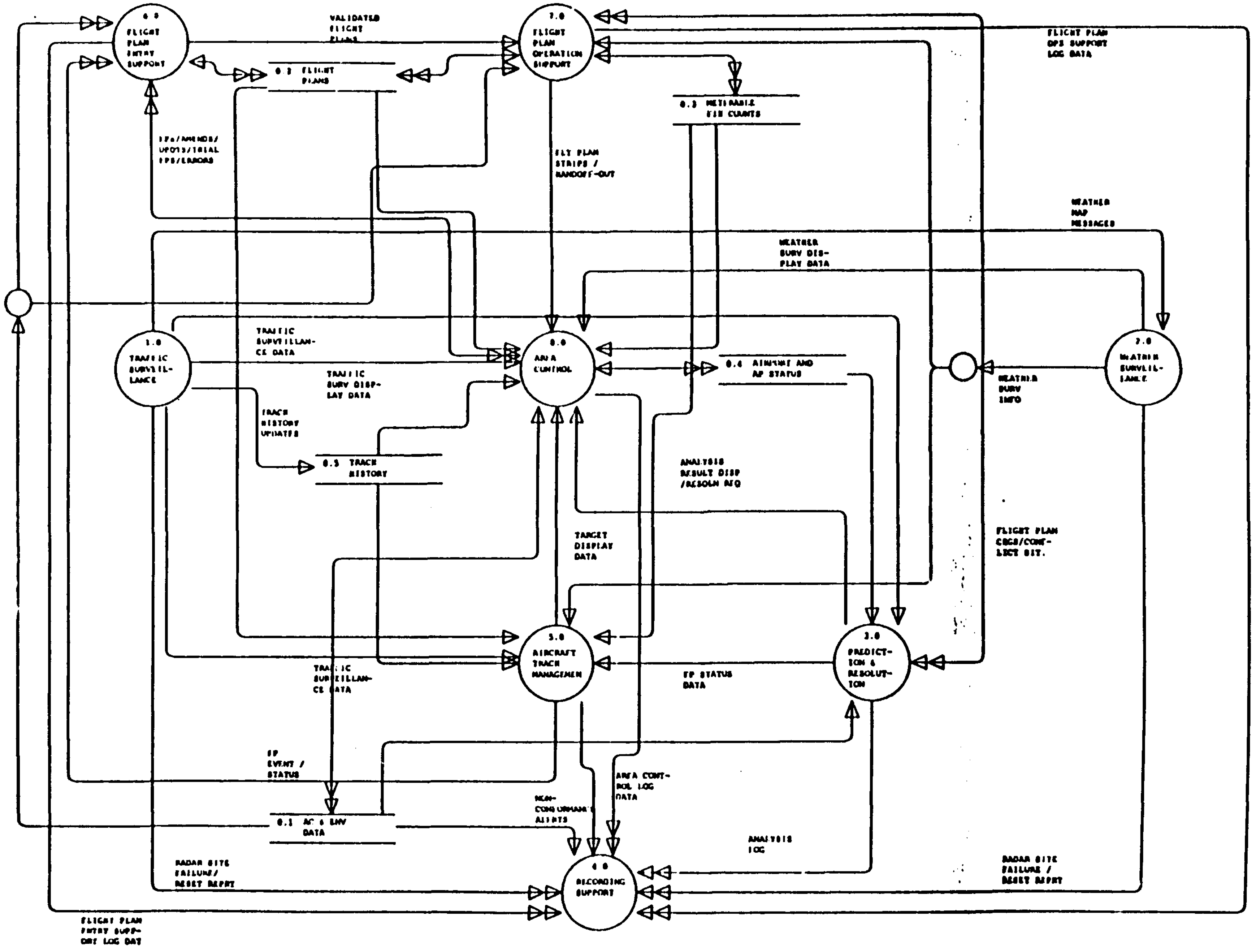

6.0 FLIGHT PLAN ENTRY SUPPORT
VALIDATED FLIGHT PLANS
7.0 FLIGHT PLAN OPERATION SUPPORT
FLIGHT PLAN OPS SUPPORT LOG DATA
0.2 FLIGHT PLANS
0.3 METERABLE FIX COUNTS
FPs/AMENDS/UPDTS/TRIAL FPS/ERRORS
FLT PLAN STRIPS / HANDOFF-OUT
WEATHER MAP MESSAGES
WEATHER SURV DISPLAY DATA
1.0 TRAFFIC SURVEILLANCE
TRAFFIC SURVEILLANCE DATA
TRAFFIC SURV DISPLAY DATA
8.0 AREA CONTROL
0.4 AIRPORT AND AP STATUS
WEATHER SURV INFO
2.0 WEATHER SURVEILLANCE
TRACK HISTORY UPDATES
0.5 TRACK HISTORY
ANALYSIS RESULT DISP /RESOLN REQ
TARGET DISPLAY DATA
FLIGHT PLAN CROSS/CONFLICT DET.
5.0 AIRCRAFT TRACK MANAGEMENT
3.0 PREDICTION & RESOLUTION
TRAFFIC SURVEILLANCE DATA
FP STATUS DATA
FP EVENT / STATUS
AREA CONTROL LOG DATA
NON-CONFORMANCE ALERTS
0.1 AC & ENV DATA
ANALYSIS LOG
RADAR SITE FAILURE/ RESET REPRT
4.0 RECORDING SUPPORT
RADAR SITE FAILURE / RESET REPRT
FLIGHT PLAN ENTRY SUPPORT LOG DAT

Figure 8. DoD AAS Area Control Facility. This figure illustrates the [illegible] functional objects of the DoD AAS Area Control System. The figure also illustrates the information [illegible]between these objects.

"VIEW-FROM" DIAGRAM

See "View-From" diagram

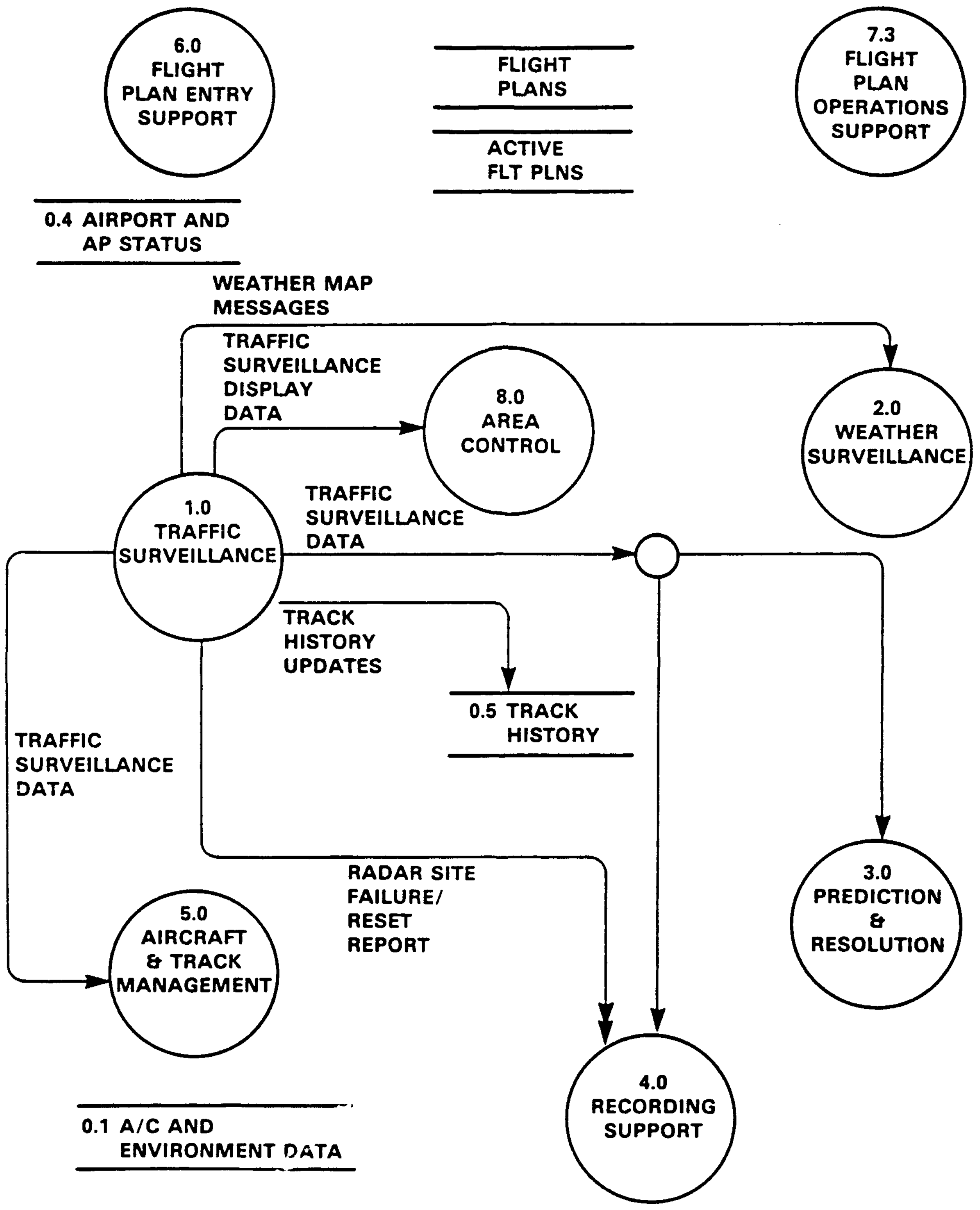

6.0 FLIGHT PLAN ENTRY SUPPORT
FLIGHT PLANS
ACTIVE FLT PLNS
7.3 FLIGHT PLAN OPERATIONS SUPPORT
0.4 AIRPORT AND AP STATUS
WEATHER MAP MESSAGES
TRAFFIC SURVEILLANCE DISPLAY DATA
8.0 AREA CONTROL
2.0 WEATHER SURVEILLANCE
TRAFFIC SURVEILLANCE DATA
1.0 TRAFFIC SURVEILLANCE
TRACK HISTORY UPDATES
0.5 TRACK HISTORY
TRAFFIC SURVEILLANCE DATA
RADAR SITE FAILURE/ RESET REPORT
5.0 AIRCRAFT & TRACK MANAGEMENT
3.0 PREDICTION & RESOLUTION
4.0 RECORDING SUPPORT
0.1 A/C AND ENVIRONMENT DATA

Figure 9. DoD AAS Area Control View from Traffic Surveillance. This figure illustrates the view of DoD AAS Area Control with respect to TRAFFIC SURVEILLANCE. This diagram also presents all of the major functional objects of the DoD AAS Area Control IOM and the message "pipes" that connect them to TRAFFIC SURVEILLANCE.

"INTERFACES" DIAGRAM

See "Interfaces" diagram

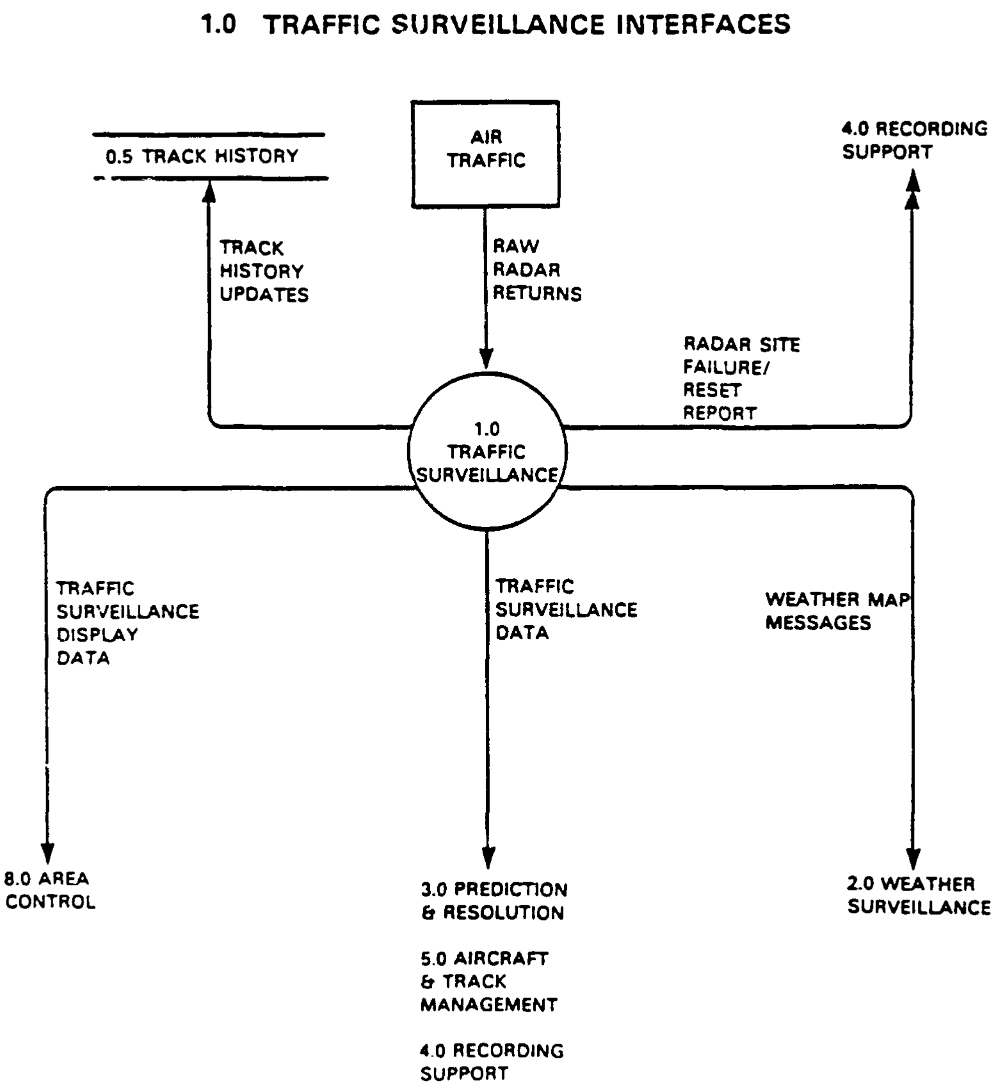
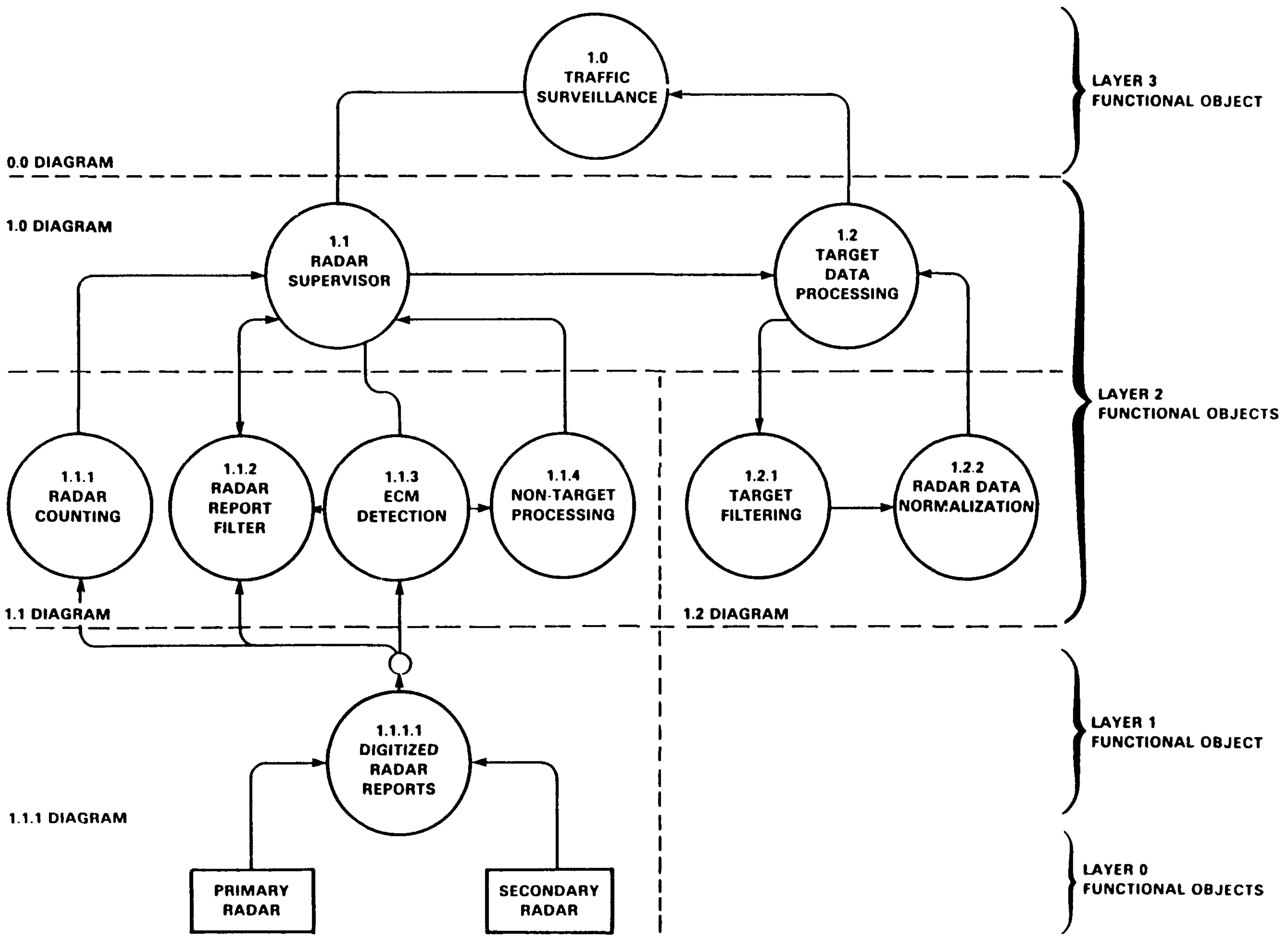


Figure 11. Interfaces for 1.0 TRAFFIC SURVEILLANCE. This figure illustrates the interfaces of the the 1.0 Traffic Surveillance functional object. This diagram shows the major inputs from other DoD AAS Area Control functional objects and external interfaces.

IDENTIFY AND ALLOCATE SUBORDINATE FOs TO EACH MFO

See "Functional Object Tree"

1.0 Traffic Surveillance Functional Object Tree



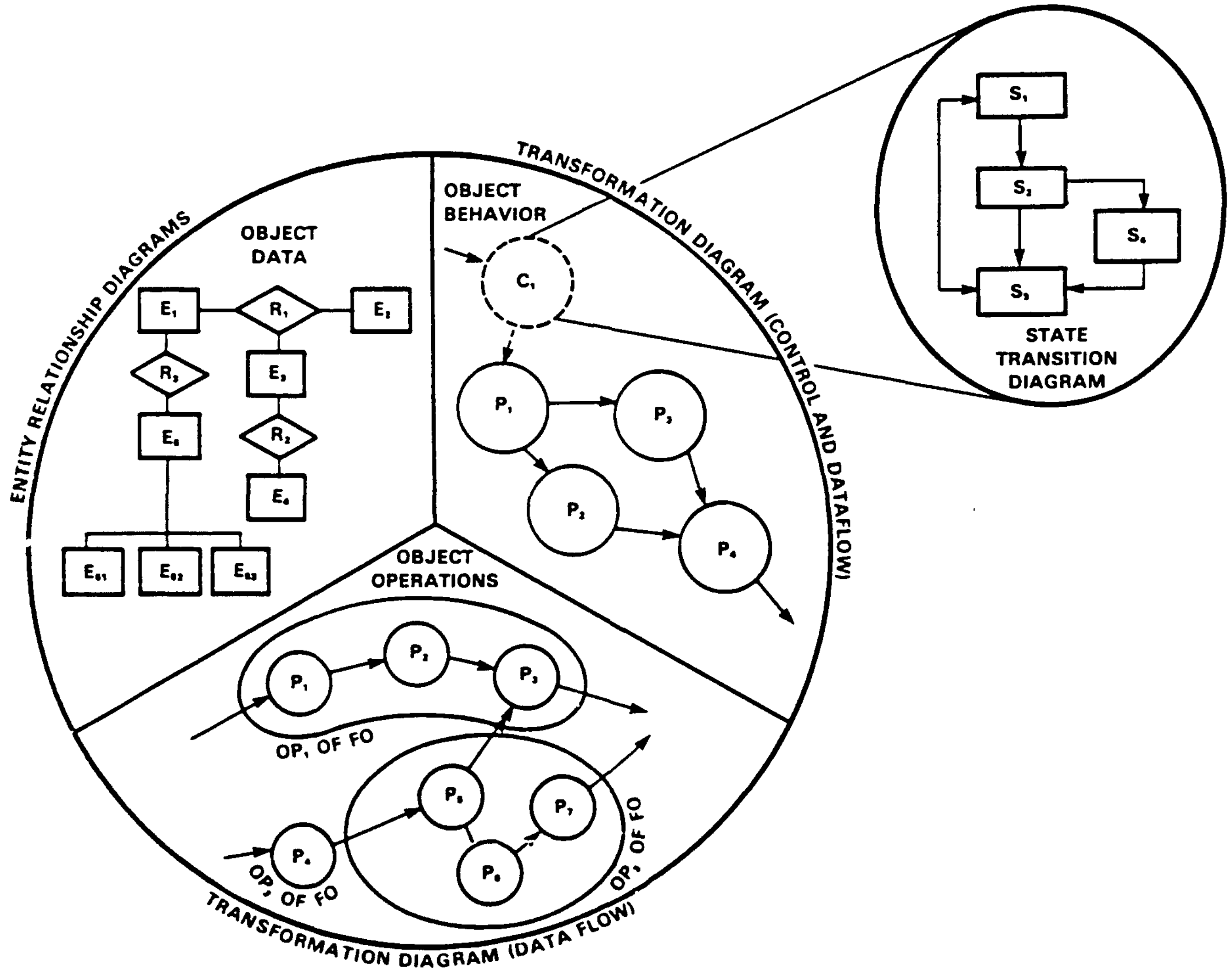
MODEL THE SYSTEM ASPECTS OF FOs AT EACH DIAGRAM LEVEL

- Data flow between peer FOs with the same parent FO
- Data flow between parent and child FOs
- Data stores employed by FOs
- Control flow between peer FOs with the same parent FO
- Control flow between parent and child FOs
- FO behavior (state transition diagrams)
- Data flows, data stores are decomposed and described

MODELING WHAT'S INSIDE EACH FUNCTIONAL OBJECT

Operations, data and behavior of a functional object can be described by:

- Textual descriptions
- Graphics (See diagram on next page).


ENTITY RELATIONSHIP DIAGRAMS
OBJECT DATA
E₁
R₁
E₂
R₃
E₃
E₅
R₂
E₄
E₅₁
E₅₂
E₅₃
TRANSFORMATION DIAGRAM (CONTROL AND DATAFLOW)
OBJECT BEHAVIOR
C₁
P₁
P₃
P₂
P₄
S₁
S₂
S₄
S₃
STATE TRANSITION DIAGRAM
OBJECT OPERATIONS
P₁
P₂
P₃
OP₁ OF FO
P₄
P₅
P₆
P₇
OP₂ OF FO
OP₃ OF FO
TRANSFORMATION DIAGRAM (DATA FLOW)

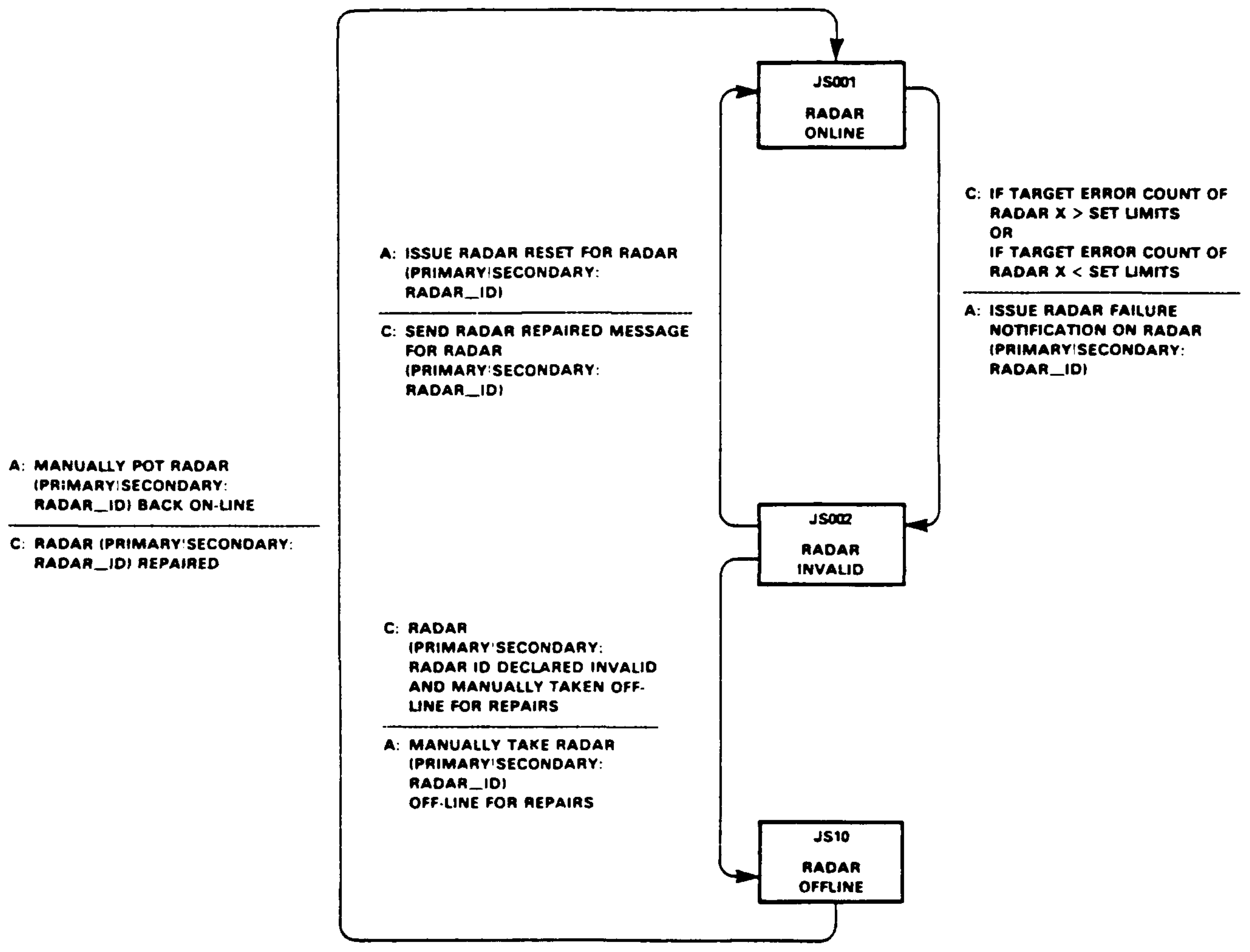

JS001
RADAR ONLINE
JS002
RADAR INVALID
JS10
RADAR OFFLINE
C: IF TARGET ERROR COUNT OF RADAR X > SET LIMITS OR IF TARGET ERROR COUNT OF RADAR X < SET LIMITS
A: ISSUE RADAR FAILURE NOTIFICATION ON RADAR (PRIMARY|SECONDARY: RADAR_ID)
A: ISSUE RADAR RESET FOR RADAR (PRIMARY|SECONDARY: RADAR_ID)
C: SEND RADAR REPAIRED MESSAGE FOR RADAR (PRIMARY'SECONDARY: RADAR_ID)
A: MANUALLY POT RADAR (PRIMARY|SECONDARY: RADAR_ID) BACK ON-LINE
C: RADAR (PRIMARY'SECONDARY: RADAR_ID) REPAIRED
C: RADAR (PRIMARY'SECONDARY: RADAR ID DECLARED INVALID AND MANUALLY TAKEN OFF-LINE FOR REPAIRS
A: MANUALLY TAKE RADAR (PRIMARY'SECONDARY: RADAR_ID) OFF-LINE FOR REPAIRS

Figure 11. 1.1 RADAR SUPERVISOR BEHAVIOR. This figure illustrates the state transition diagram for changing between three states, namely: RADAR ONLINE, RADAR INVALID, and RADAR OFFLINE.

MODEL MESSAGE COMMUNICATION BETWEEN FOs BETWEEN IOM LAYERS

- Model message communication between IOM layers (and diagram levels):
  - Data flow between parent and child FOs
  - Control flow between parent and child FOs

DATA FLOW BETWEEN PARENT AND CHILD FO

See diagram on next page

IOM FORM EXAMPLE

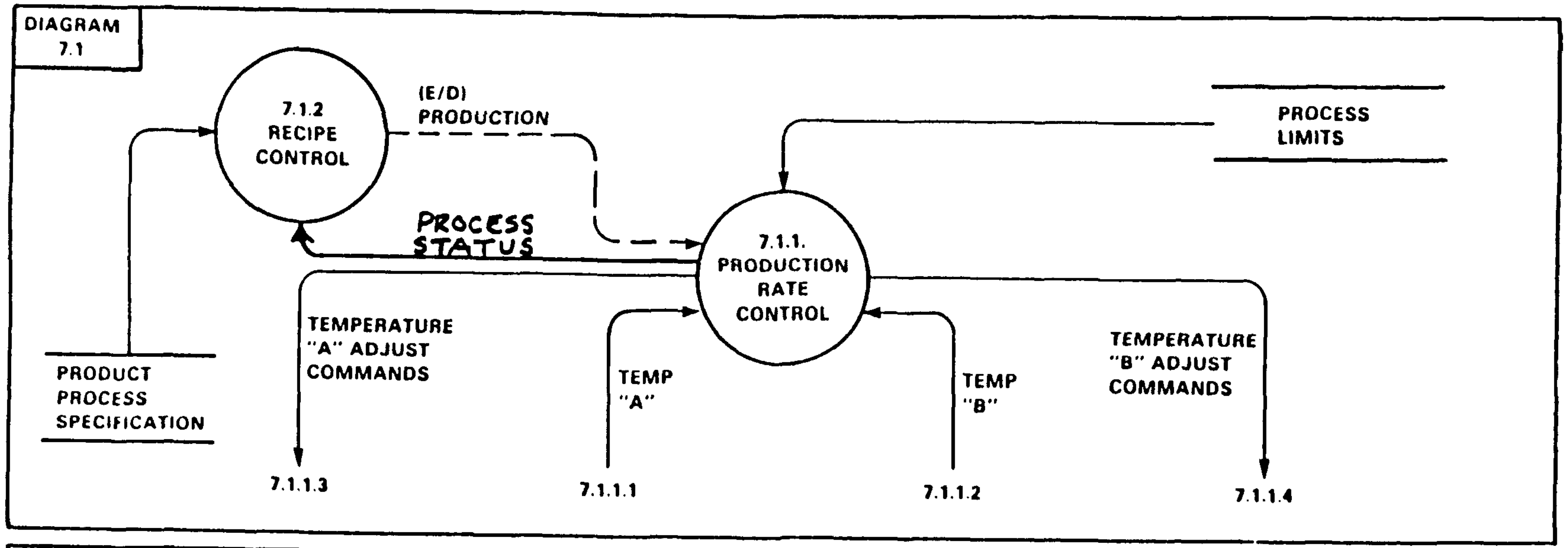
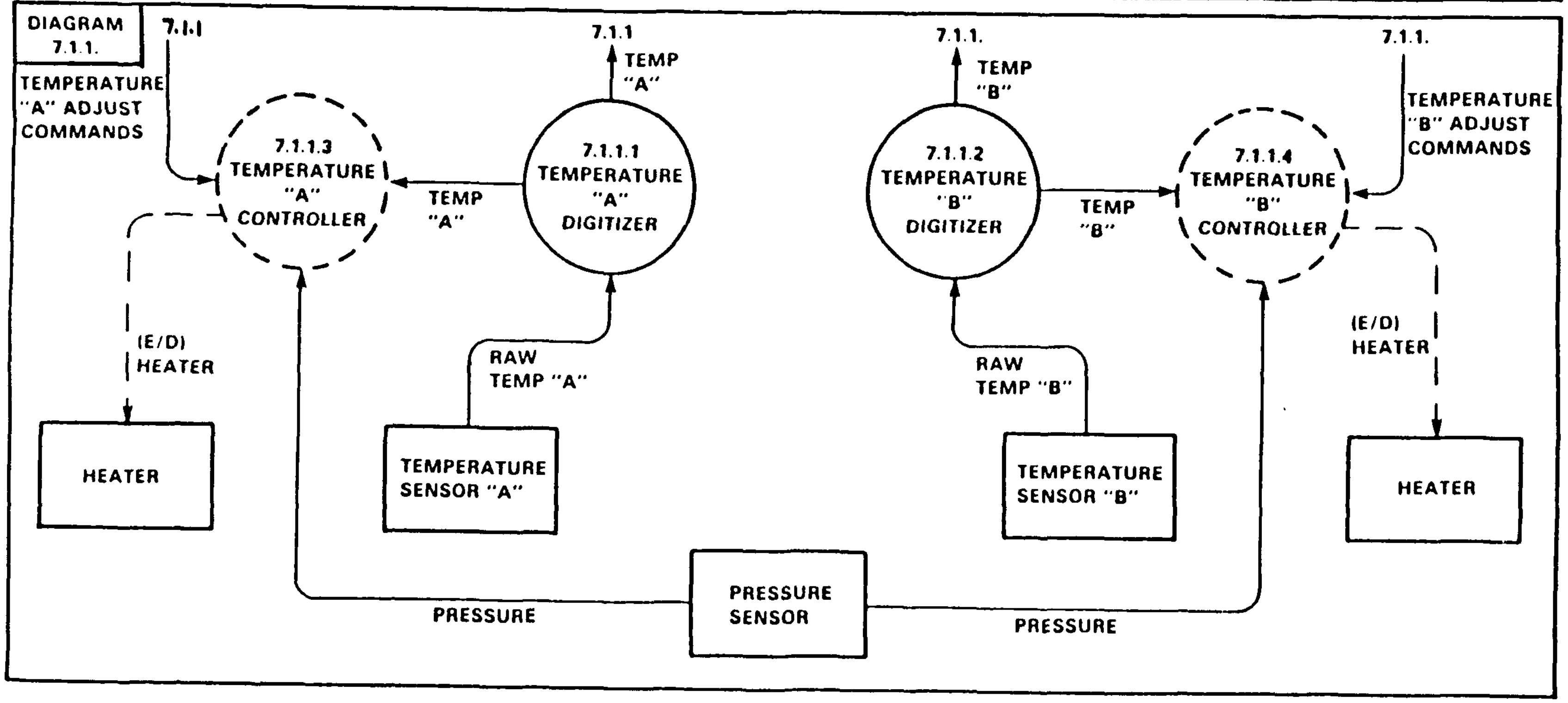


DIAGRAM 7.1
7.1.2 RECIPE CONTROL
(E/D) PRODUCTION
PROCESS LIMITS
PROCESS STATUS
7.1.1. PRODUCTION RATE CONTROL
PRODUCT PROCESS SPECIFICATION
TEMPERATURE "A" ADJUST COMMANDS
TEMP "A"
TEMP "B"
TEMPERATURE "B" ADJUST COMMANDS
7.1.1.3
7.1.1.1
7.1.1.2
7.1.1.4



CONTROL FLOW BETWEEN PARENT AND CHILD FO

See diagram on next page

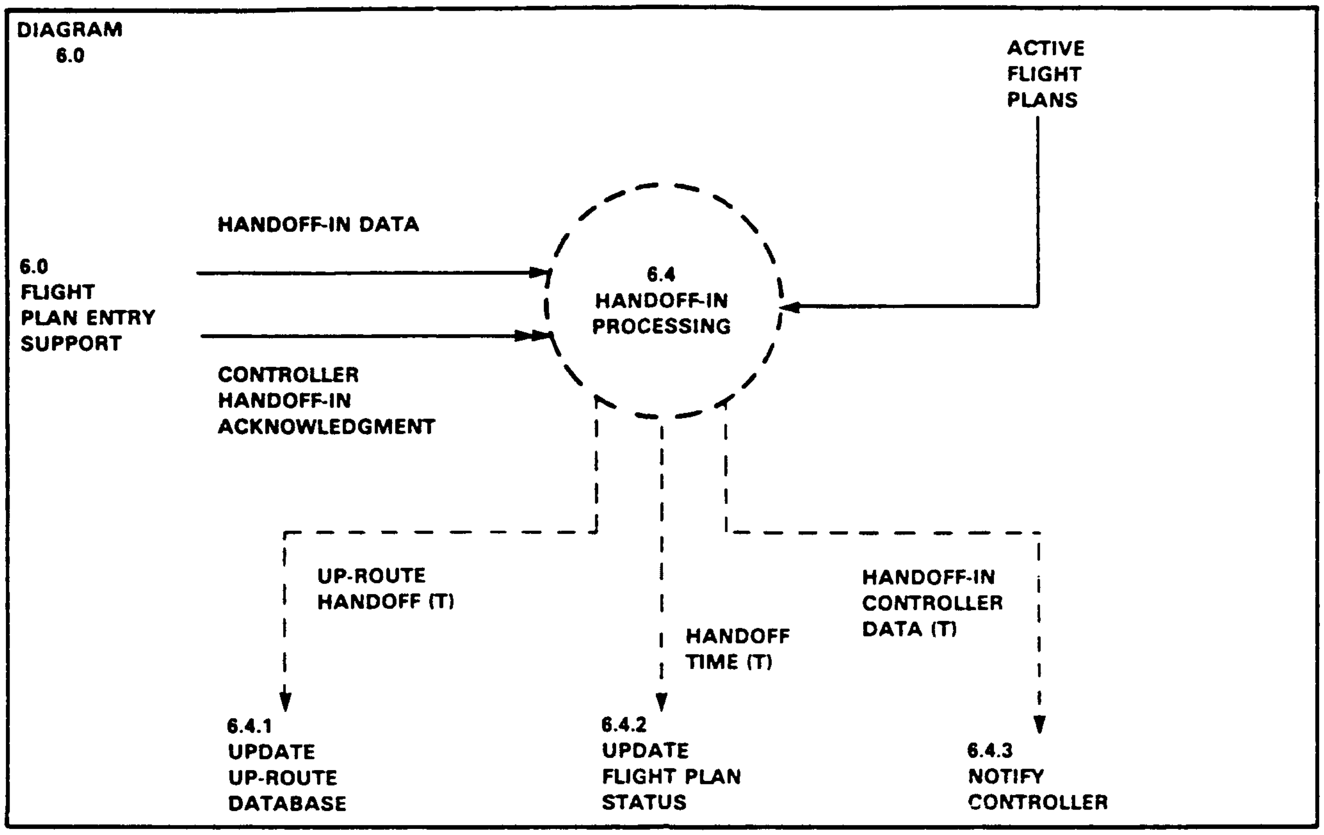


DIAGRAM 6.0
ACTIVE FLIGHT PLANS
HANDOFF-IN DATA
6.0 FLIGHT PLAN ENTRY SUPPORT
6.4 HANDOFF-IN PROCESSING
CONTROLLER HANDOFF-IN ACKNOWLEDGMENT
UP-ROUTE HANDOFF (T)
HANDOFF TIME (T)
HANDOFF-IN CONTROLLER DATA (T)
6.4.1 UPDATE UP-ROUTE DATABASE
6.4.2 UPDATE FLIGHT PLAN STATUS
6.4.3 NOTIFY CONTROLLER

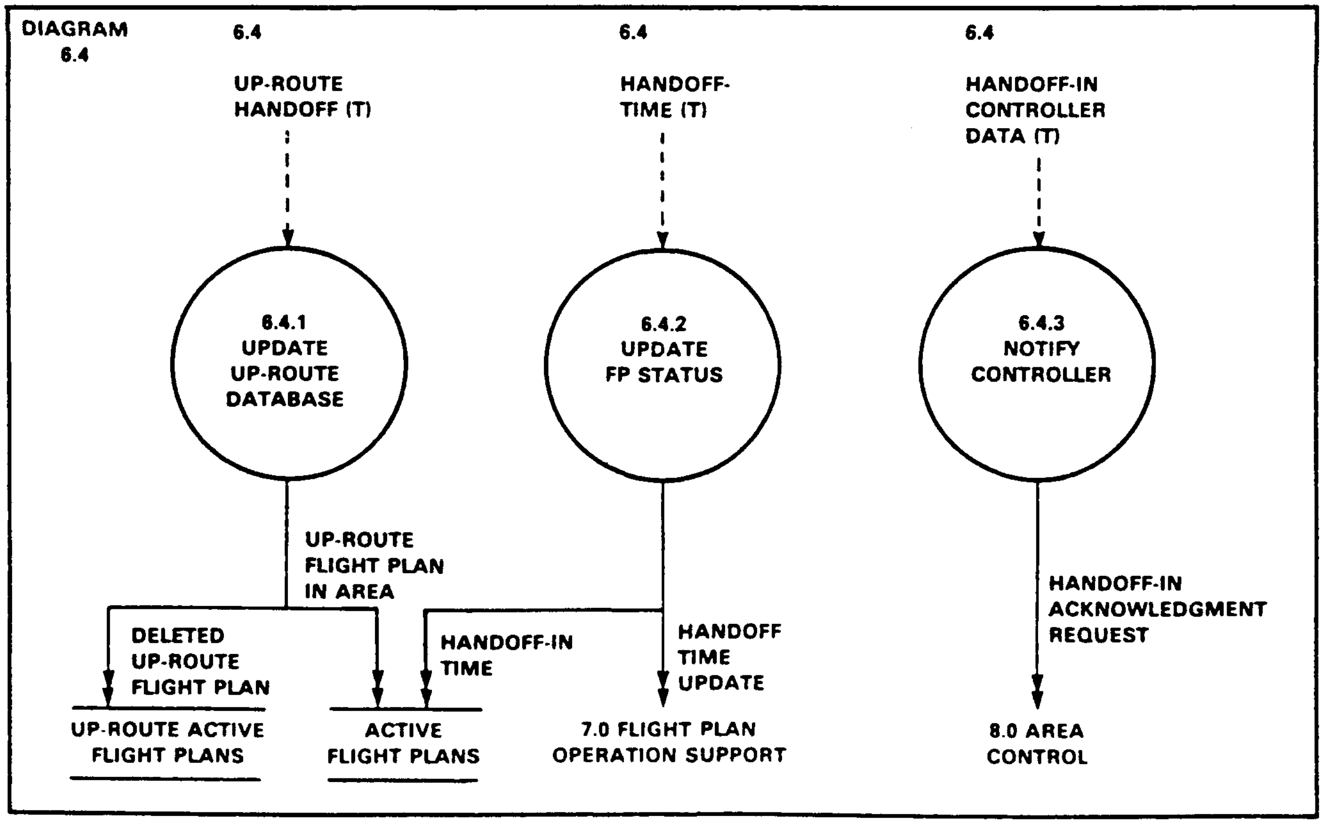


DIAGRAM 6.4
6.4 UP-ROUTE HANDOFF (T)
6.4 HANDOFF-TIME (T)
6.4 HANDOFF-IN CONTROLLER DATA (T)
6.4.1 UPDATE UP-ROUTE DATABASE
6.4.2 UPDATE FP STATUS
6.4.3 NOTIFY CONTROLLER
UP-ROUTE FLIGHT PLAN IN AREA
DELETED UP-ROUTE FLIGHT PLAN
HANDOFF-IN TIME
HANDOFF TIME UPDATE
HANDOFF-IN ACKNOWLEDGMENT REQUEST
UP-ROUTE ACTIVE FLIGHT PLANS
ACTIVE FLIGHT PLANS
7.0 FLIGHT PLAN OPERATION SUPPORT
8.0 AREA CONTROL

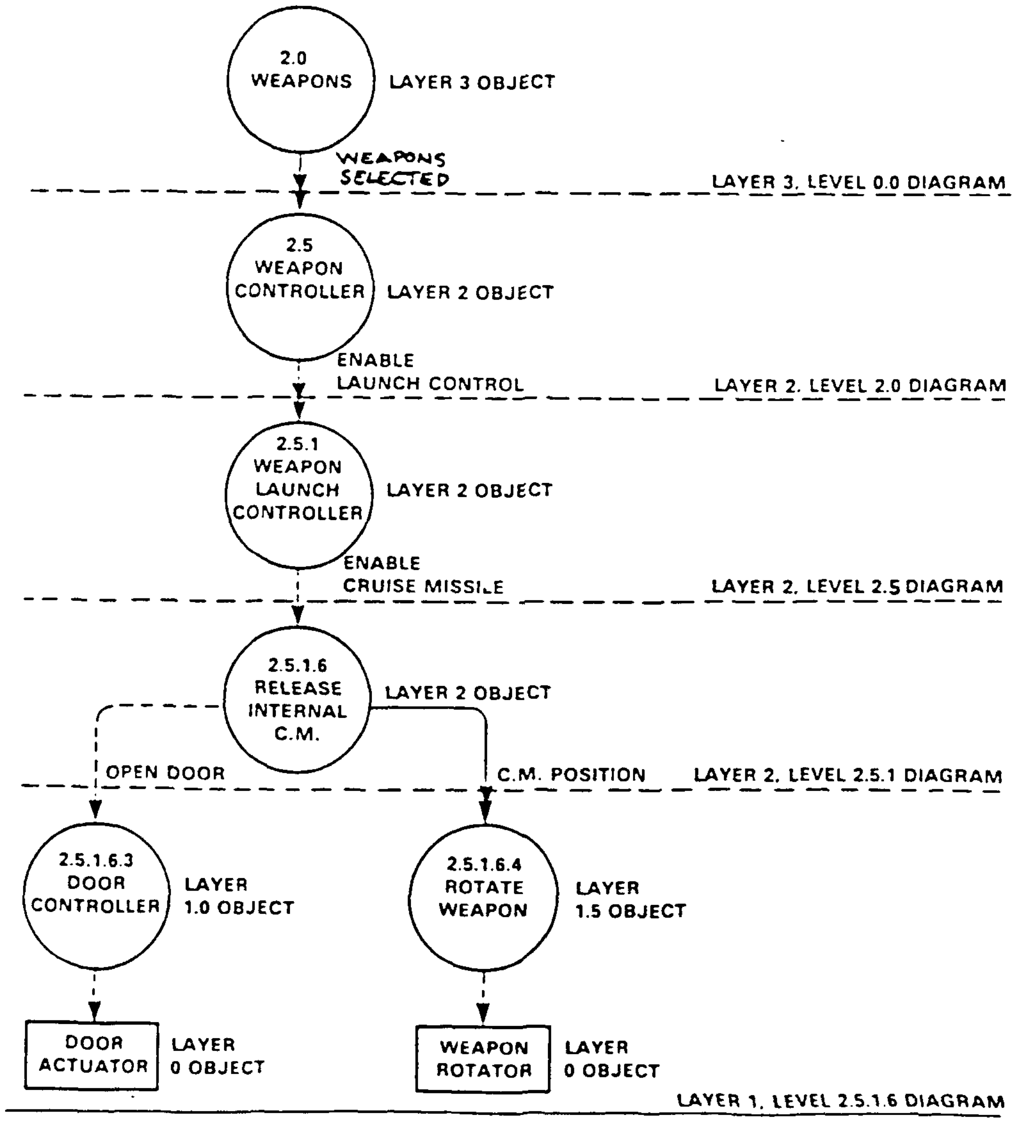


Figure 18. IOM Layers versus Levels. This figure illustrates an IOM hierarchical thread from the 2.0 Weapons Functional Object, which shows communication between IOM layers and levels. Also note that an IOM layer can comprise several levels.

REVIEW, REFINEMENT AND ACCEPTANCE

- IOM first-pass book is presented to and reviewed by the customer
- Comments, questions and concerns are discussed and addressed
- IOM second-pass book is presented to and reviewed by the customer
- IOM is either accepted, or refined/updated as necessary

SUPPORTING DIAGRAMS

- IOM Layer versus diagram level
- IOM Communication
- IOM Communication problems
- Non-Communicating peers
- "Hollow Bubbles"

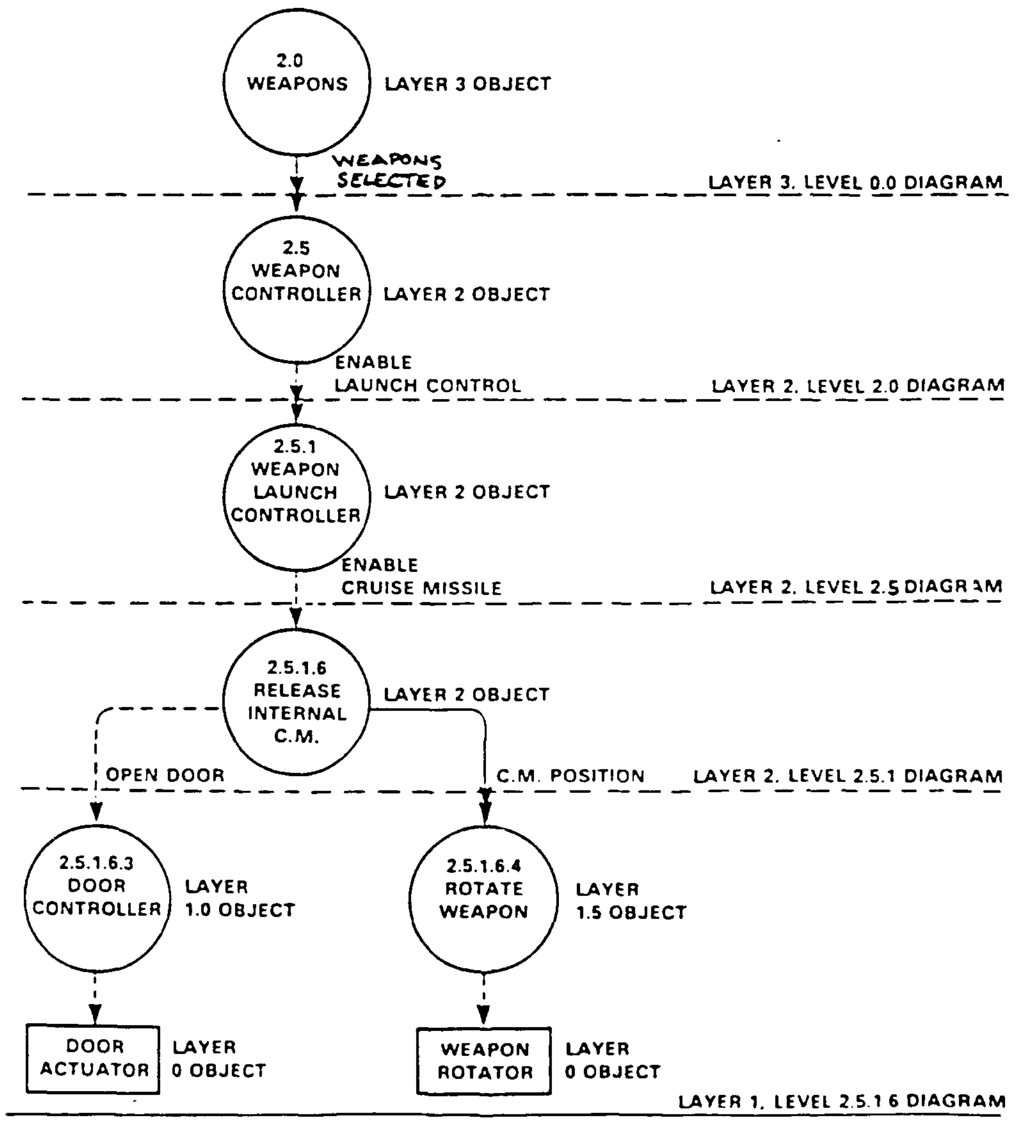


Figure 18. IOM Layers versus Levels. This figure illustrates an IOM hierarchical thread from the 2.0 Weapons Functional Object, which shows communication between IOM layers and levels. Also note that an IOM layer can comprise several levels.

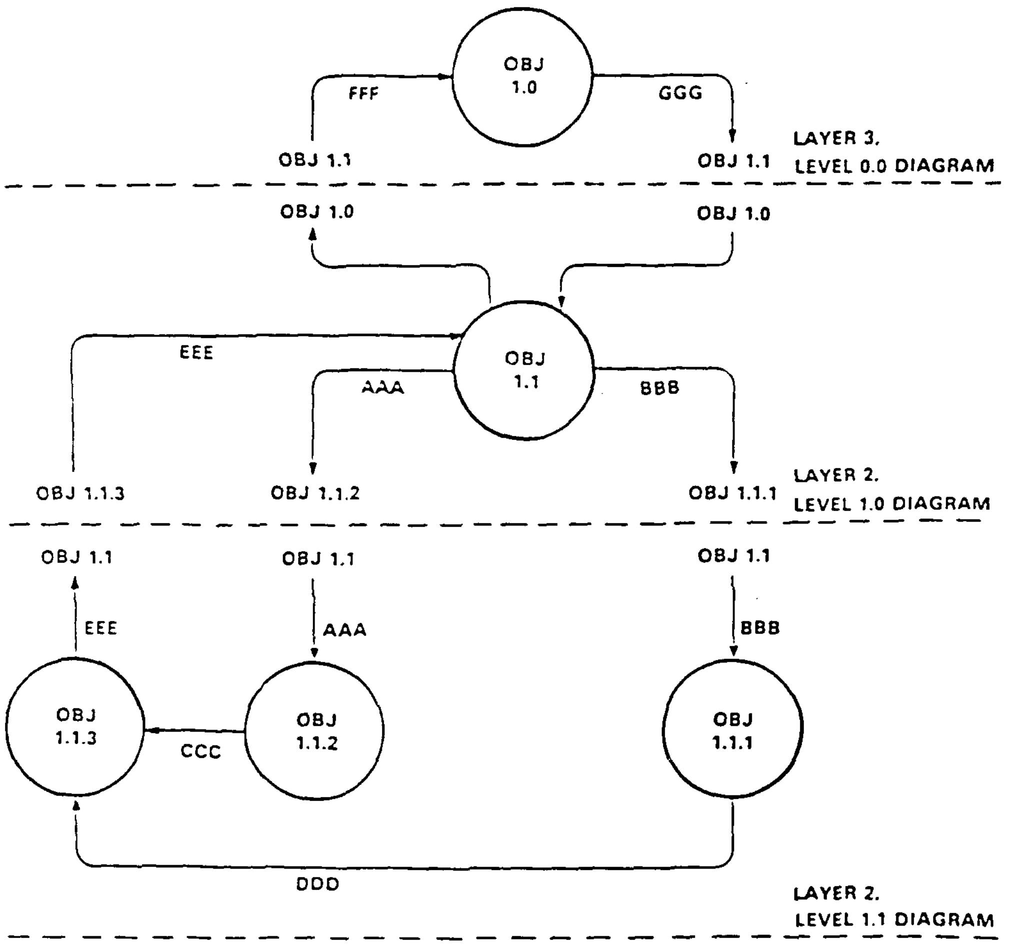


Figure 17. IOM Object Communication: Parent, Child, Peer. This figure illustrates that all functional objects should have at least two of the three forms of communication: 1) communication (data control) between peer objects with the same parent, 2) communication (data control) between a functional object and its sibling objects, and 3) communication (data control) between a functional object and its parent object.

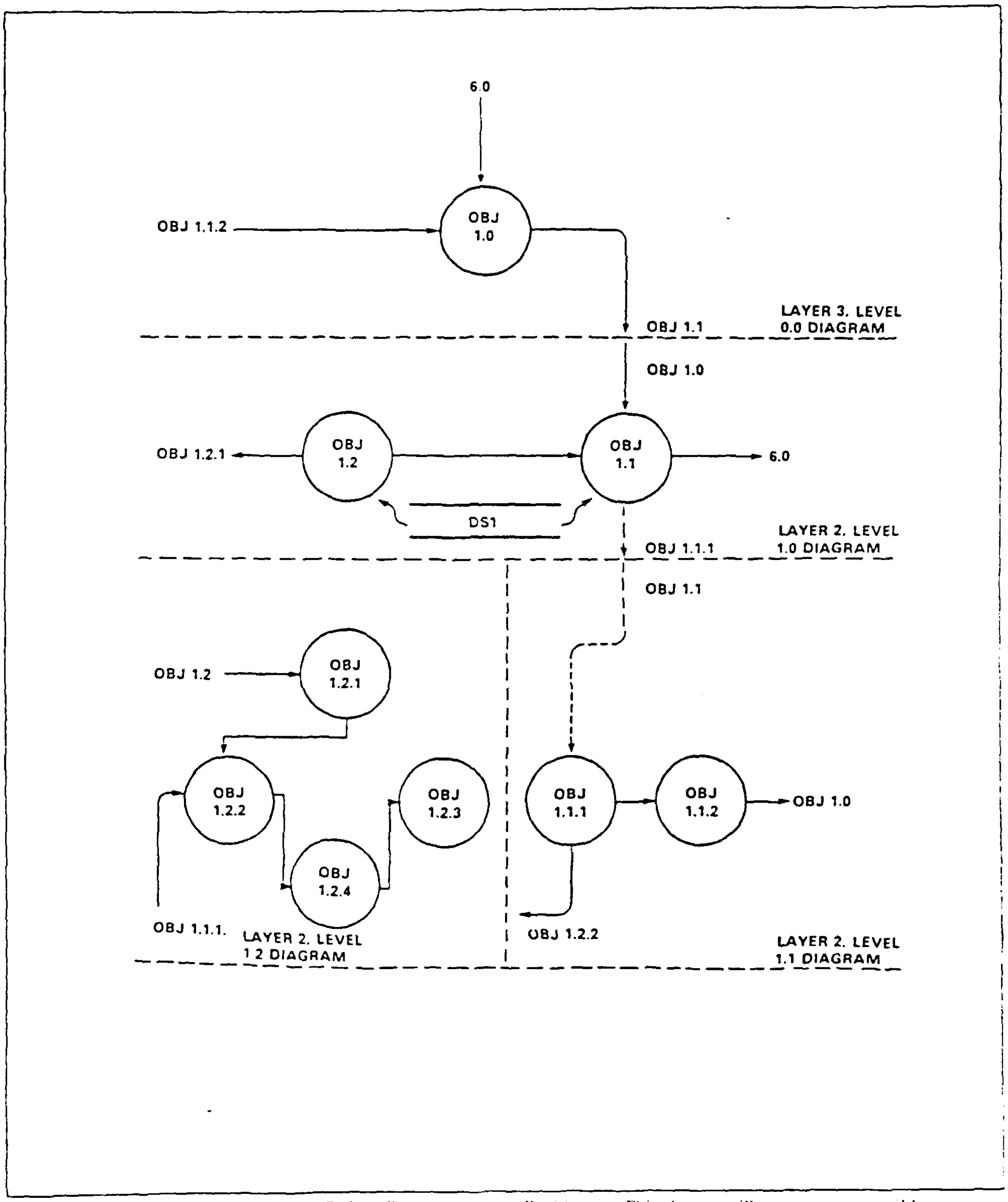


Figure 19. Possible IOM Functional Object Communication Problems. This diagram illustrates two possible communication problems, namely:

- Typical objects communicate with their peers, or with parent or child levels. There may be a problem with object 1.1.2, which directly sends data to object 1.0, and thus bypass its parent diagram object
- Object 1.1.1 communicates with object 1.2.2: in an IOM, cousin objects should not communicate.

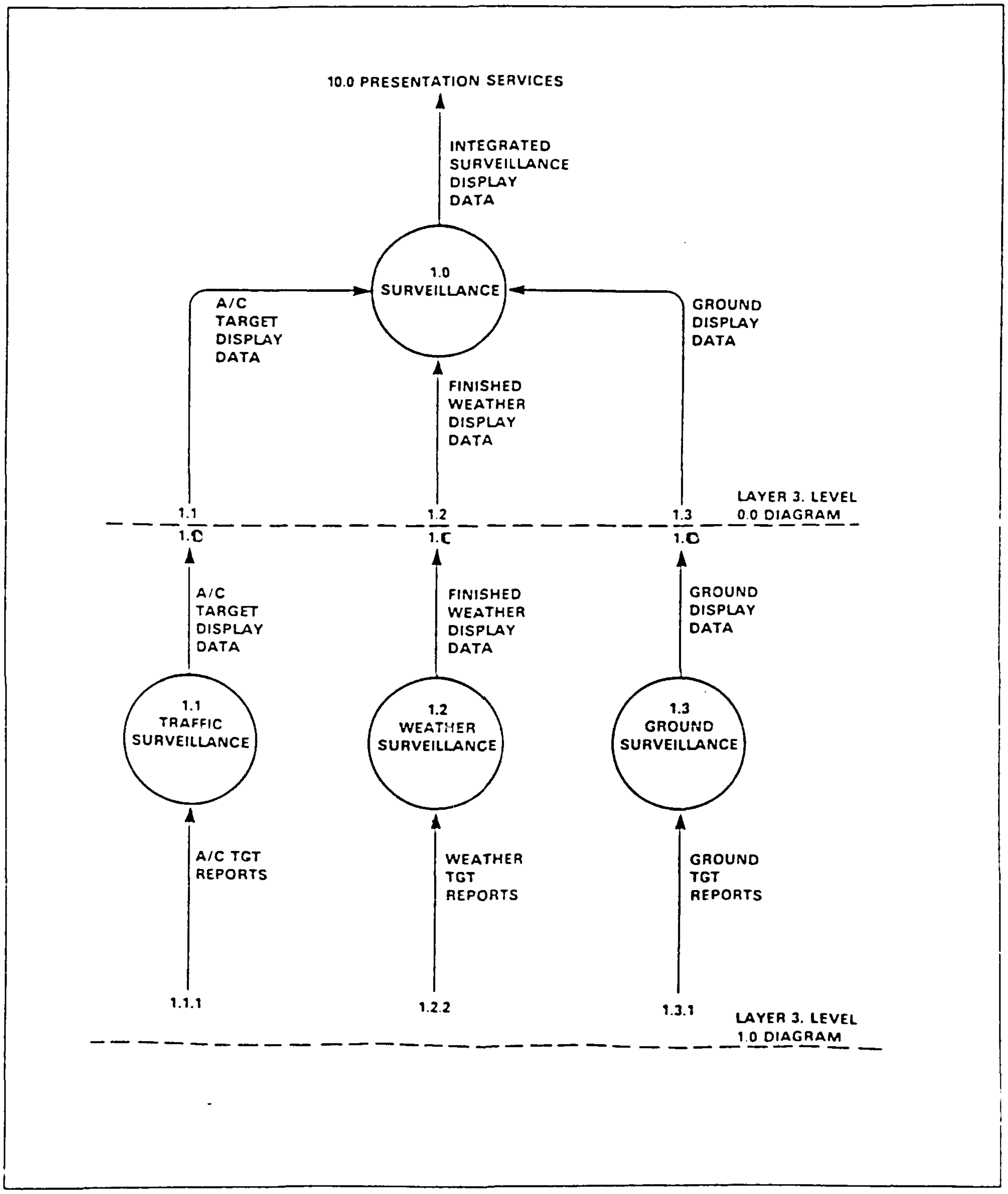


Figure 20. Non-Communication Peer Functional Objects. This figure represents a perfectly reasonable IOM drawing, where functional objects illustrate communication between child and parent drawing objects. However, 1.1, 1.2 and 1.3 do not communicate with each other. There is no peer to peer data or control communication. 1.1, 1.2 and 1.3 may need to be promoted or coalesced.

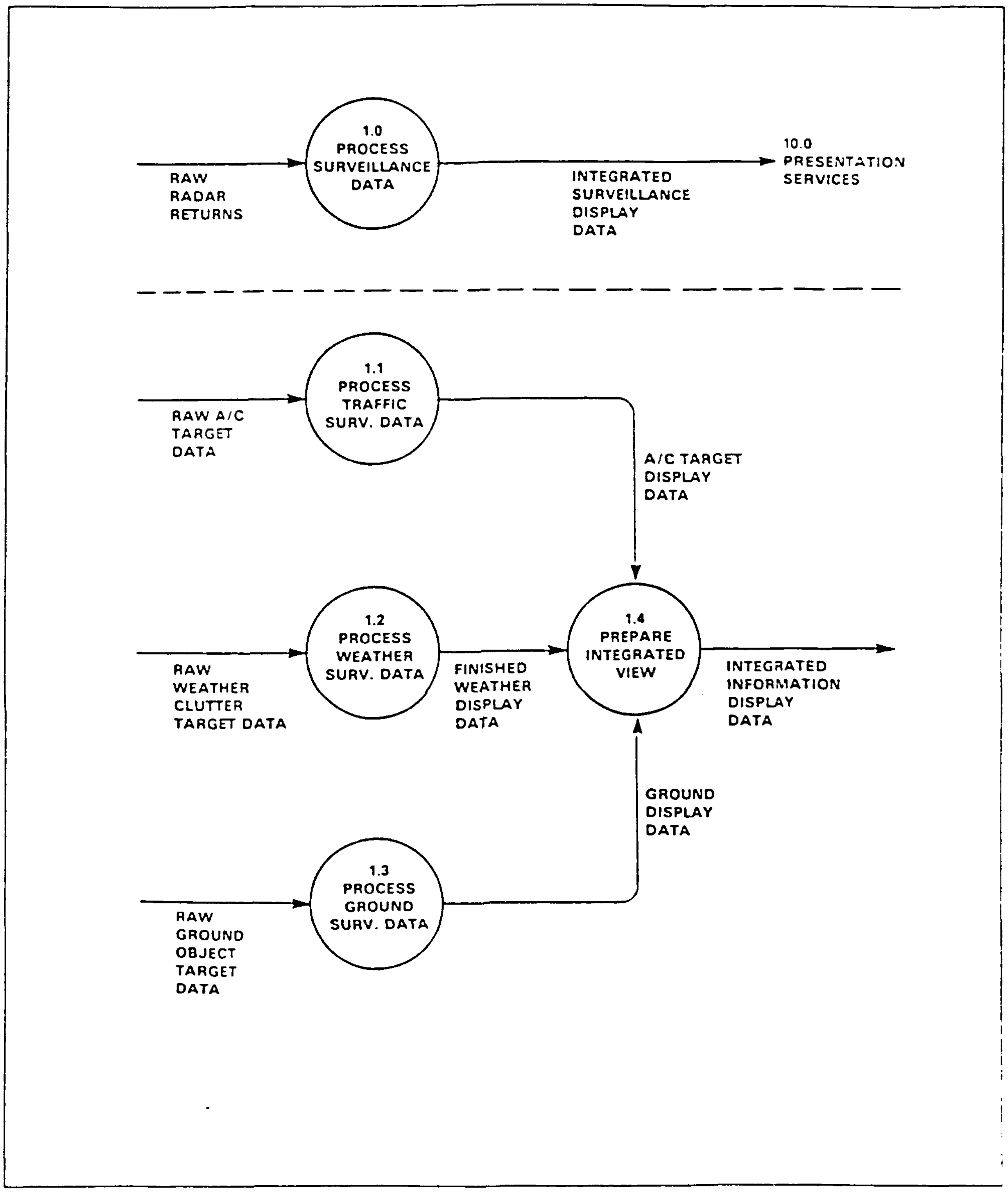


Figure 21. "Hollow Bubbles" Portraying Functional Objects. This diagram illustrates 1.0 as a "hollow bubble." 1.0 represents a packaging of processes for convenience and may obscure the fact that a reorganization of functional objects may be desirable. Also note that this diagram, which represents a structured analysis view of "1.0 PROCESS SURVEILLANCE DATA," does not highlight message communication as an IOM representation would.

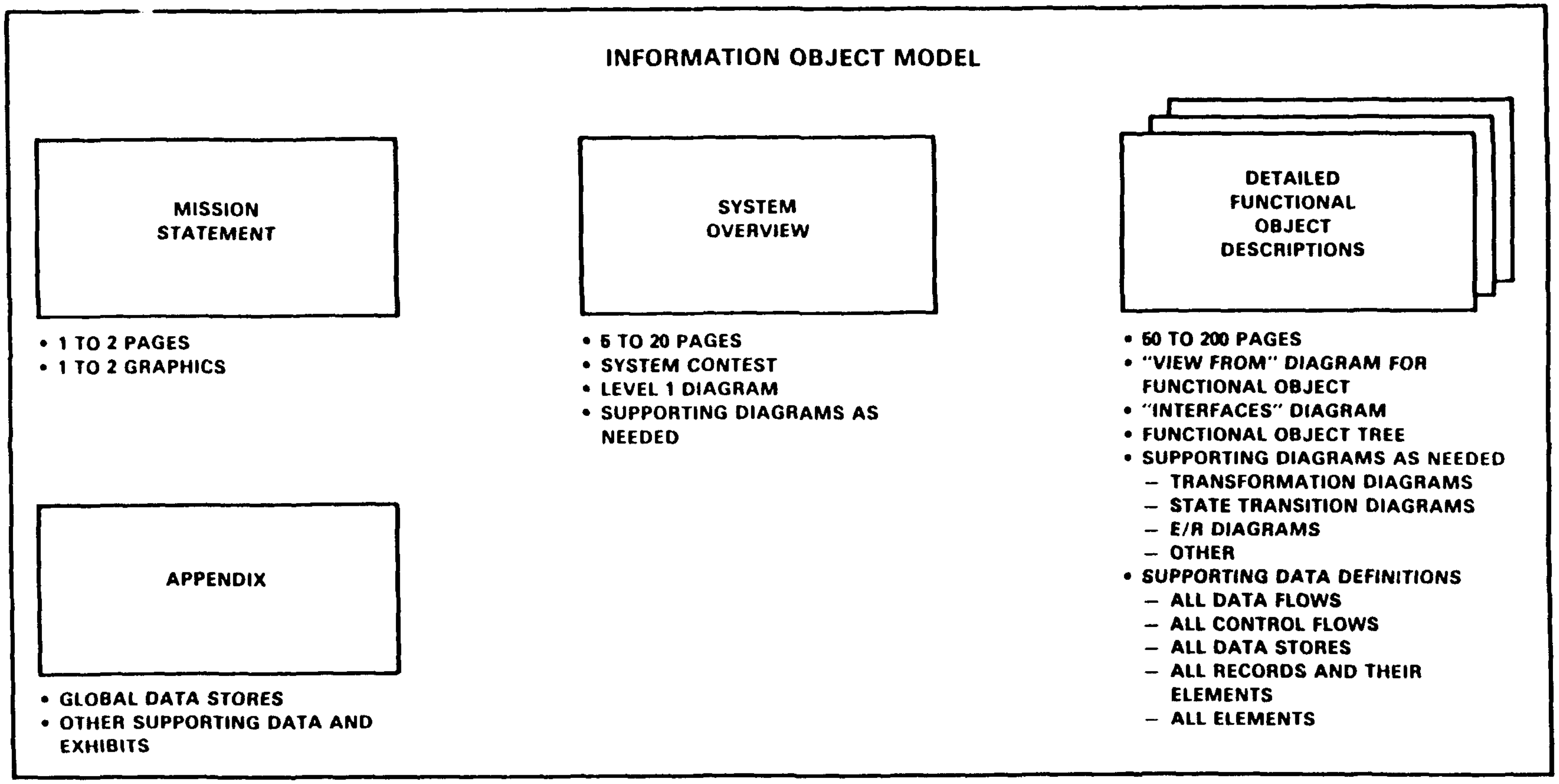
PACKAGING THE IOM

- Introduction
  - Task information
  - Task scoping rationale
  - Recommendations, issues and concerns
- Mission Statement
  - Problem statement
  - Operational concept statement
- System Overview
  - Expansion of mission statement
  - Overall system description
  - Describes how the MFOs identified satisfy the mission statement
  - Introduced by:
    - System Context Diagram (level-0)
    - Level-1 Diagram

PACKAGING THE IOM (Cont.)

- Major Functional Object Detail Descriptions
  - Detailed description of the MFOs
  - Detailed descriptions of the subordinate FOs
  - Description for each FO identified includes:
    - Level I+1 diagram description
    - Description of FOs in each diagram
    - Description of the inputs and outputs for each FO
    - State transition diagram, where required
    - description of the FOs operations in PDL
    - Summary data for MFO
      - All data flows
      - All data stores
      - All records and their elements
      - All elements
      - All control flows
      - All control transforms
- Appendix
  - Other information and reports relevant to the IOM
    - Global Data Store Descriptions
    - CASE-produced reports

IOM Packaging Concept



WHAT CAN YOU DO WITH THE INFORMATION OBJECT MODEL?

- The IOM can used as a specification for:
  - Foxboro-style Implementation Model development
  - Ward-Mellor-style Implementation Model development
  - Incrementally developing system prototypes
- The IOM could serve as a procurement tool:
  - Assist DoD Program Managers understand a complex system
  - Help to identify technology breakthroughs required
  - Help to understand how a procurement might be approached

THE IOM'S ROLE IN THE SYSTEMS DEVELOPMENT LIFE CYCLE

- Potential role in the reuse-drive systems development life cycle
- Potential role in the Software-First Life Cycle
- Potential role in the Spiral Model of system development

POTENTIAL ROLE IN THE REUSE-DRIVEN SYSTEMS DEVELOPMENT LIFE CYCLE

- Twin life cycles in the reuse-driven SDLC
  - Domain-specific generic products development
  - Application-specific system development

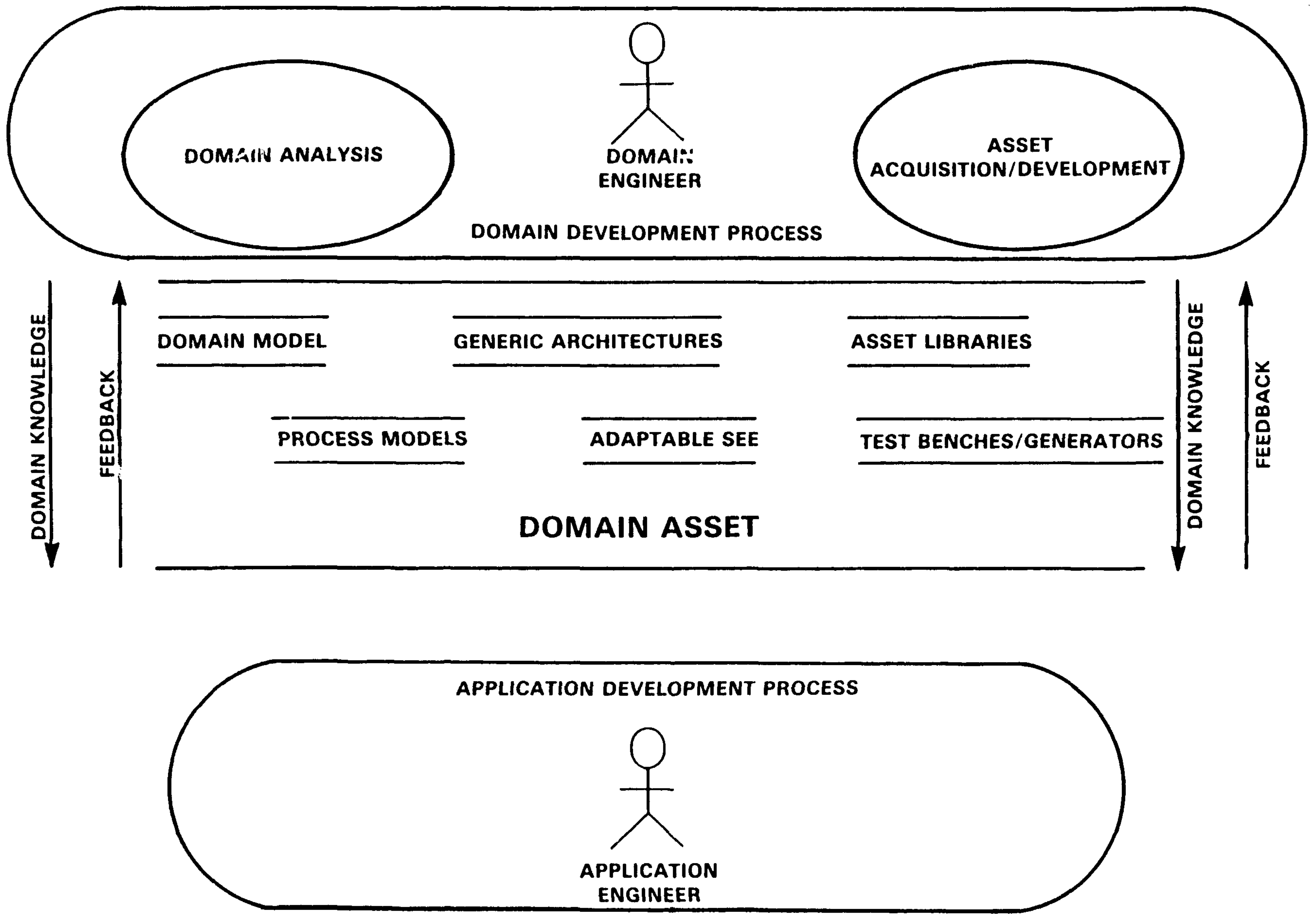
- Domain-specific generic products may include:
  - Generic IOMs for common problems identified in a given problem domain (reuse quality)

- Application-specific systems development
  - Develop generic IOM to get a better understanding of the problem domain (not reuse quality)
  - Develop application-specific IOM to specify a target system

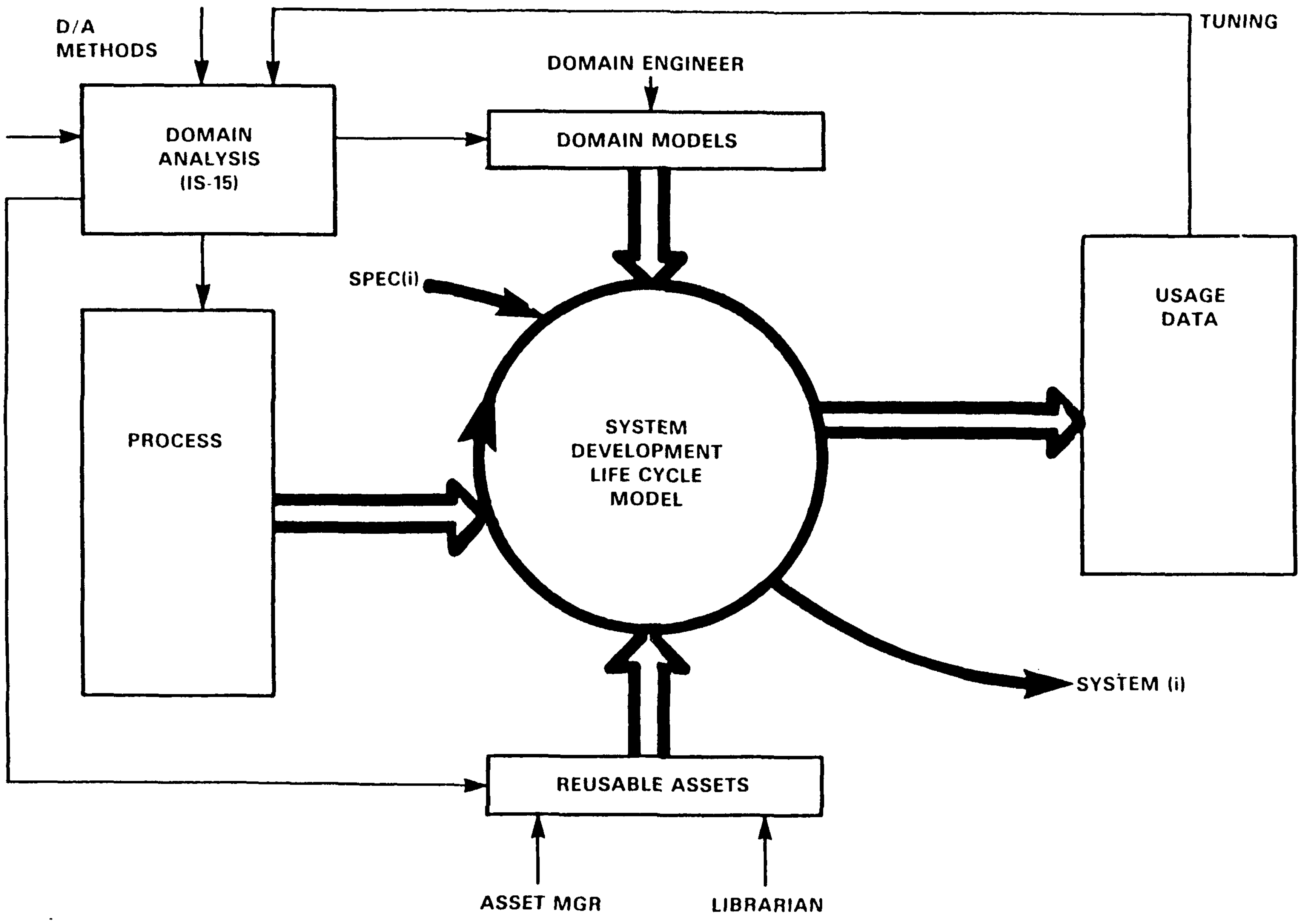
REUSE-DRIVEN SYSTEMS DEVELOPMENT LIFE CYCLE

See Figure Next Page

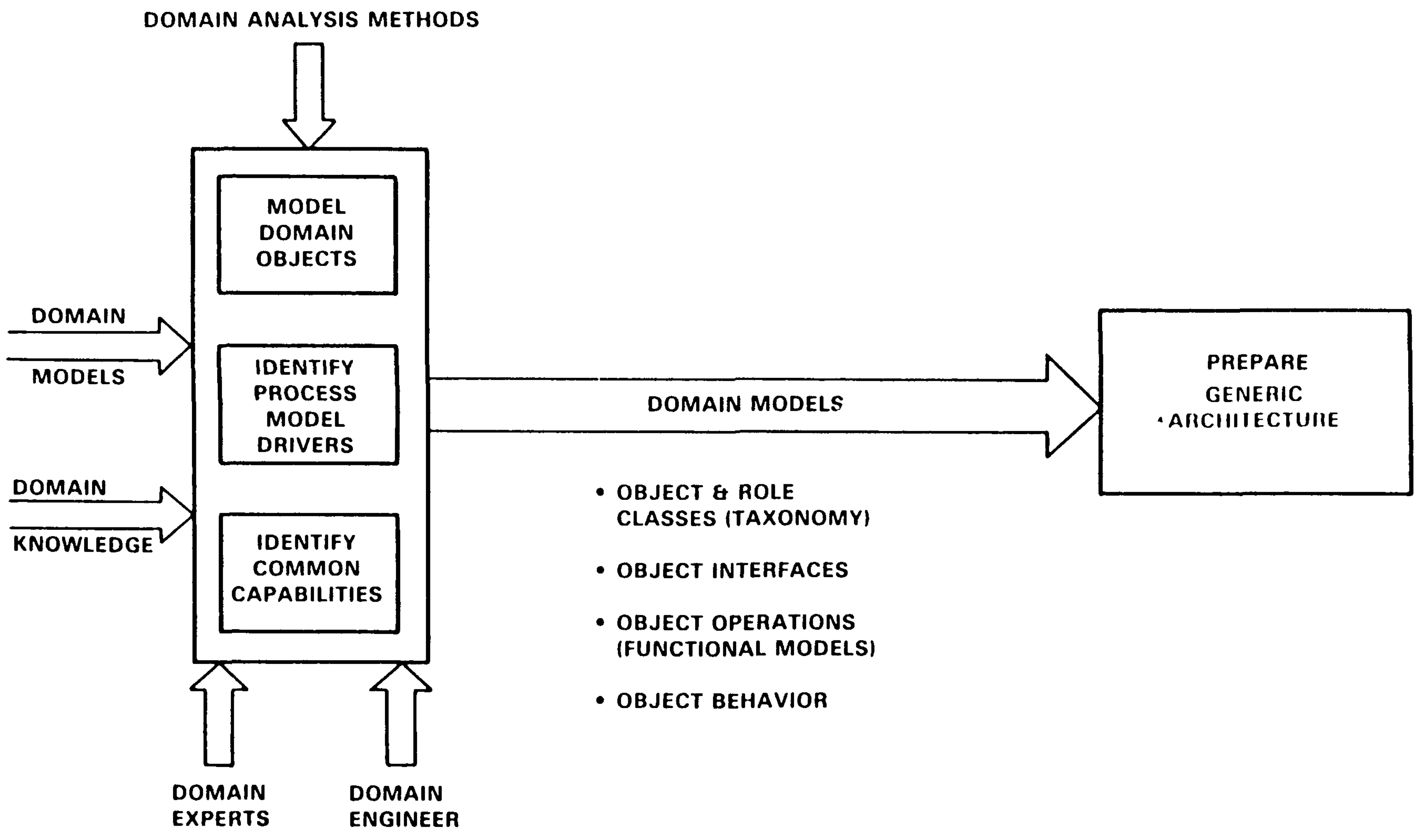
IBM TEAM APPLICATION DEVELOPMENT MODEL TWO CONCURRENT LIFE CYCLES



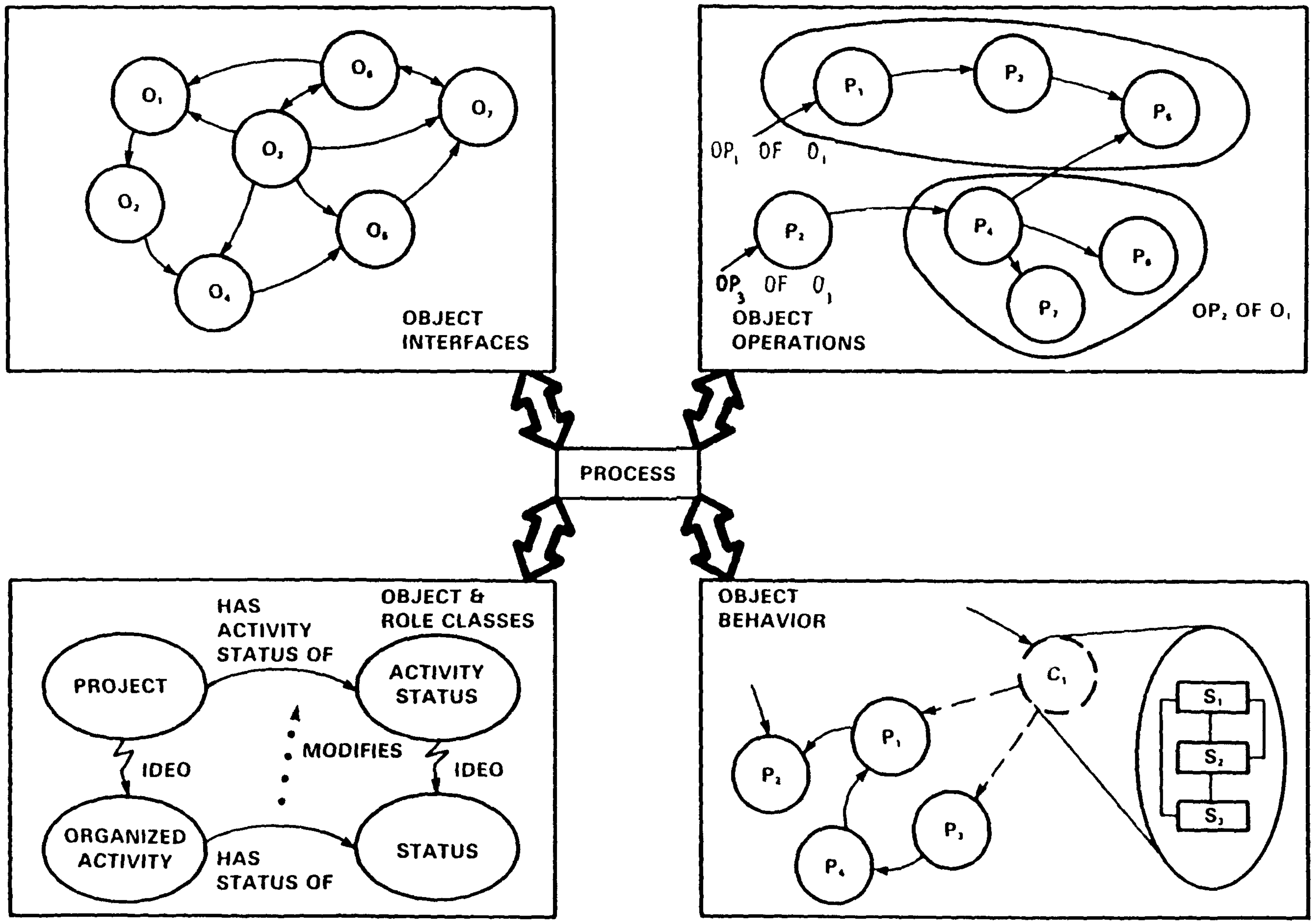
IS-15: REUSE IN THE SOFTWARE DEVELOPMENT PROCESS



IS-15.2: ANALYZE SEE DOMAIN


DOMAIN ANALYSIS METHODS
MODEL DOMAIN OBJECTS
IDENTIFY PROCESS MODEL DRIVERS
IDENTIFY COMMON CAPABILITIES
DOMAIN MODELS
DOMAIN KNOWLEDGE
DOMAIN EXPERTS
DOMAIN ENGINEER
DOMAIN MODELS
PREPARE GENERIC ARCHITECTURE
• OBJECT & ROLE CLASSES (TAXONOMY)
• OBJECT INTERFACES
• OBJECT OPERATIONS (FUNCTIONAL MODELS)
• OBJECT BEHAVIOR

IS-15.2: SEE DOMAIN MODEL



POTENTIAL ROLE IN THE SOFTWARE-FIRST LIFE CYCLE

- Information Object Modeling could be performed during "Preliminary Systems Analysis"

- Specification produced could be used to conduct the "Pre-Prototype Review"

- IOM can be incrementally updated during the life cycle as necessary; becomes final before "Productization and Production"

SOFTWARE-FIRST LIFE CYCLE - CONCEPTUAL VIEW

```
    | Desired Operational
    |    Capability
   \|/
+--------+               +---------+               +----------+  +-------
|Prelim. |  ...........  | System  |  ..........   | Product- |  |System
|System  |  .Pre-     .  | Archi-  |  .PreProd-.   | ization  |  |Operati
|Analysis|->.Prototyp-.->| tecture |->.uctiz-  .-> |    &     |->| and
|        |  .ing      .  |         |  .ation   .   |Production|  |Support
|        |  .Review   .  |         |  .Review  .   |          |  |
+--------+  ...........  +-+---+---+  ..........   +----------+  +----+--
/|\ /|\ |                  |   | /|\                        /|\  /|\  |
 |   |  |                  |   |  |                          |    |   |
 |   |  |  New Technology  |   |  |                          |    |   |
 |   |  |     Requests     |   |  +--------------+           |    |   |
 |   |  +----------------->|   |                 |           |    |   |
 |   |                     |   |                 |           |    |   |
 |  \|/                    |   |                \|/         \|/  \|/  |
 |  /----------\ . . . . . . . . . . . . .  /----------\  . . . . . . |
 |  |Repository|           |   |            |Repository|              |
 |  +----------+ . . . . . . . . . . . . .  +----------+  . . . . . . |
 |      |                  |   |                /|\                   |
 |      |          New     |   Component         |                    |
 |      |          Technology  Development       |                    |
 |      |          Requests|   Requests          |                    |
 |      |                 \|/ \|/                |                    |
 |      |                 +---------+            |                    |
 |      +---------------->|         |<-----------+                    |
 |                        | Software|                                 |
 |                        | Growing |                                 |
 |                        +---------+                                 |
 |                                        New Operational Capability  |
 +--------------------------------------------------------------------+
```

POTENTIAL ROLE IN THE SPIRAL MODEL OF SYSTEMS DEVELOPMENT

- Incrementally develop an IOM during each spiral cycle:

  - Establish objectives for a portion of the product to be developed - (Mission Statement)

  - Develop a sufficient mini-IOM for the identified product portion

  - Identify candidate methods for implementing or reusing FOs identified in IOM

  - Identify risks associated with each candidate

LESSONS LEARNED AND CONCLUSIONS

- Information Object Modeling methodology strengths:
  - Partially supports object-orientation
  - Fosters a more terse specification model than yielded by a RTSA
  - Works well for problems that are amenable to hierarchic control as a solution
  - Specifications can probably be produced in 90 days given an:
    - Understanding objects of the problem domain
    - Understanding the problem domain with respect to the target system
  - Produces a comprehensive specification model

- The Information Object Modeling methodology weaknesses:
  - It is not a general purpose modeling methodology
    - It is difficult to apply to data-driven information systems
  - At Foxboro - it is a level 2 process - understood and practiced by experts, but not documented

WHAT WE LEARNED ABOUT FOXBORO

- Experts in the domain of process control

- They practice reuse:
  - 46 basic objects for integrating process control loops
  - Objects are in firmware
  - When objects are required they are developed in "C", entered in an "object library" and managed by an object manager
  - Object manager mediates system execution for both stand-alone and distributed processing

WHAT WE LEARNED ABOUT FOXBORO (CONT.)

- Foxboro understands their domain:
  - Foxboro has domain models of process control
  - Foxboro understands their objects (H/W, S/W and F/W)
  - Foxboro can specify complex systems using the Information Modeling Methodology in 90 days
    - Identify appropriate process control paradigm to apply
    - Identify where there are exceptions
    - Identify problems and risks
    - Does not accept a job unless there is at least 65% Foxboro content

CONCLUSIONS

- The Information Object Modeling methodology is no silver bullet
- The key to more rapidly developing specifications are:
  - Having access to domain models for the target problem domain
  - Understanding the objects of ihat domain
  - Being able to allocate objects given a template of capabilities for that domain, e.g., the "layered model"

CONCLUSIONS (Cont.)

- The Information Object Modeling methodology could be reasonably applied for DoD systems-in-the-large given:

Analysis planning effort	30 days
Team orientation	5 days
Methodology and tools training	25 days
A cursory problem domain analysis (Generic IOM)	45 days
Problem Specific IOM development	120 days
Information Object Modeling Effort	225 days

ASSERTIONS TO BE PROVED

- A team can produce a specification for a complex DoD system in 90 days
  - Yes, giving the following assumptions are true:
    - An adequate analysis team is available
    - The team leader is not assigned as one of the analysts
    - The team is fully trained in the methodology
    - The team has an understanding of the problem domain and its objects
    - The target problem can be modeled using the hierarchic control paradigm

- The specification produced can be used as the entry point for "Software Growing" of the Software-First Systems Life Cycle
  - Although this has never been tried, it seems reasonable

- The Information Object Model techniques are "object-oriented"
  - The techniques require you model the processing required with functional objects, where each object has:
    - Has encapsulated operations
    - Has encapsulated data
  - One could view functional objects as "actors"